2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION
Paper ID #50667

RelSim: A Process-Based Synthetic Relational Database Generator for Engineering Education

Yan Fang, University of Southern California
Dr. Bruce Wilcox, University of Southern California

Dr. Wilcox is a senior analytics consultant with over 30 years years experience with top-tier consulting firms providing management and information systems consulting services to large corporate and government clients. From 2013 until 2021, he was employed full-time by the SAS Institute, a premier provider of advanced analytics software and consulting services, responsible for consulting with major SAS government clients in California on the use of advanced analytics tools and solutions.

With BS and MS degrees in Electrical Engineering from Carnegie-Mellon University and an MBA from UCLA, Bruce earned his PhD in Engineering and Applied Mathematics from Claremont Graduate University in 2018 with thesis work in time series clustering for fixed income portfolio diversification using model-based clustering and state space analysis techniques. Since January 2020 until taking a full-time faculty position in the 2021 fall semester, Bruce was a part-time lecturer at USC Vitterbi College of Engineering in the MS Analytics program. Prior to that, he taught for twelve semesters as a lecturer at California State University, Long Beach and as a visiting international professor at the National Economics University, Hanoi, Vietnam, teaching two short courses in quantitative analysis to advanced finance students.

Firas K Elshaer

RelSim: A Process-Based Synthetic Relational Database Generator for Engineering Education

Yan Fang, Bruce Wilcox, Firas Elshaer
University of Southern California, Los Angeles, CA 90089, USA

Abstract

Database and data engineering instruction is often constrained by the limited availability of realistic, open-source relational datasets. Widely used examples such as Sakila and Northwind are small, static, and quickly exhausted in the classroom, limiting instructors' ability to create varied assignments across cohorts and restricting opportunities to teach realistic temporal workflows and operational provenance. This limitation is increasingly problematic in modern data science and artificial intelligence (DSAI) curricula, where students must integrate multi-table data, reason over event logs, engineer features from temporal histories, and construct model-ready datasets without information leakage.

To address this gap, we present RelSim, an open-source platform for generating synthetic relational databases and event traces using a simulation-driven, process-based approach. RelSim operates in three phases: Configure, where instructors define schemas and workflows using a concise declarative specification; Generate, where entity populations are created using templates, formulas, and distributions; and Simulate, where discrete-event process logic produces time-stamped operational records that naturally reflect waiting, capacity constraints, and causal ordering. Unlike purely statistical synthetic data generators, RelSim produces relational coherence and temporal consistency by construction.

This paper describes the design motivation and architecture of RelSim and demonstrates its capabilities through a healthcare scheduling case study. We also present validation results showing referential integrity and temporal consistency across generated tables, including many-to-many relationships. Finally, we outline a planned Spring 2026 classroom deployment in a master's-level analytics engineering course and define an evaluation framework to assess learning outcomes related to multi-table data integration, temporal reasoning, and feature engineering in DSAI workflows.

Keywords: database education, synthetic database, discrete-event simulation, SQL practice, open-source datasets, YAML configuration

1. INTRODUCTION

Relational databases remain foundational in modern analytics, data engineering, and artificial intelligence systems. Although many educational programs teach SQL and relational modeling using small example databases, these datasets often fail to reflect the structure and complexity of real operational environments. Instructors frequently rely on widely known open-source datasets such as Sakila or Northwind, which provide useful schema examples but are limited in size, scope, and realism. Because these datasets are static and quickly become familiar to students,

they are difficult to reuse across multiple cohorts and provide limited opportunity for varied assignments, project-based learning, or scenario-driven exercises.

This limitation is increasingly significant in data science and artificial intelligence (DSAI) education. Modern workflows typically involve integrating normalized operational data, constructing features from event histories, reasoning about temporal dependencies, and building reproducible pipelines that feed downstream machine learning models. These learning objectives require datasets that contain realistic temporal behavior, operational provenance, and multi-table dependencies. Static educational databases rarely capture such dynamics, and synthetic data tools designed for single tables do not naturally extend to coherent multi-table relational environments.

An additional limitation of most static educational datasets is that they do not support realistic periodic database updates, which are essential for teaching modern analytics engineering practices such as incremental ELT pipelines and slowly changing dimension (SCD) handling. In real organizations, operational databases evolve continuously as customers, employees, projects, and other entities change over time, and analytical systems must capture these changes through snapshots, history tables, or Type 2 dimension updates. Designing instructional exercises around SCD concepts therefore requires datasets that contain coherent longitudinal history and repeatable monthly or weekly update cycles. Generating such evolving multi-table databases is challenging because updates must remain consistent across related tables, preserve temporal causality, and reflect realistic process-driven changes rather than arbitrary record edits. Supporting these periodic update scenarios is a key motivation for RelSim's simulation-based approach.

Generating realistic synthetic relational databases is substantially more difficult than generating synthetic single-table data. In a relational database, data must satisfy referential integrity constraints across tables, preserve meaningful cross-table dependencies that emerge only after joining tables, and reflect the temporal and causal ordering of operational events. Furthermore, many realistic datasets are governed by implicit business rules and domain semantics that are not fully encoded in schema constraints alone. Traditional synthetic data generation approaches often rely on statistical replication, learning distributions from existing datasets and generating new records that approximate observed patterns. While effective in some contexts, these methods typically require representative training data and may struggle to capture temporal realism, operational causality, and rare but instructionally valuable scenarios.

A recent paper [1] that developed a methodology and tool for benchmarking the fidelity and unity of relational databases concluded that the methods they evaluated "are not yet able to generate synthetic relational data that is indistinguishable from original data. Most methods have problems with marginal distributions at least on some benchmark datasets and with single tables on most datasets."

To address these limitations, we propose a complementary approach we refer to as causal generation, in which relational data is produced as the outcome of explicitly modeled operational processes. This paper introduces RelSim, an open-source simulation-driven platform that generates relational databases and event traces from declarative schema and workflow

specifications. By using discrete-event simulation as the generative mechanism, RelSim produces timestamped operational records whose structure reflects process logic, waiting, and resource constraints rather than independent sampling.

1.1 Positioning within DSAI Education

Although RelSim generates relational databases, its educational relevance extends beyond traditional database instruction. Data science and AI students must increasingly reason about operational data environments, including event traces, temporal joins, feature engineering from histories, and the construction of model-ready datasets without temporal leakage. RelSim is designed to support these learning objectives by providing configurable synthetic source systems that resemble real-world operational databases, enabling pipeline development and scenario-driven analytics assignments within DSAI curricula.

1.2 Research Questions

This work is motivated by the hypothesis that process-generated relational datasets can enable more authentic and effective instruction in data science and AI workflows. The following research questions guide the educational motivation and planned evaluation of RelSim:

RQ1. How does the use of process-generated relational datasets affect students' ability to reason about temporal ordering, causal dependencies, and data provenance in multi-table environments?

RQ2. How does student performance on multi-table data integration tasks (e.g., joining normalized tables, aggregating event histories, constructing analytical features) differ when using RelSim-generated datasets compared to traditional static datasets such as Sakila or Northwind?

RQ3. How does the availability of simulation-generated event traces influence students' ability to build model-ready datasets, particularly with respect to temporal alignment and prevention of information leakage?

RQ4. What instructional benefits does RelSim provide in supporting repeated course offerings, including the ability to regenerate dataset variants and create scenario-specific synthetic data for assignments and assessments?

RQ5. How do students perceive the realism, engagement value, and instructional usefulness of RelSim-generated datasets relative to traditional static educational databases?

1.3 Intended Learning Outcomes

RelSim is designed to support data science and AI instruction by enabling realistic multi-table datasets with operational provenance and temporal behavior. In particular, RelSim is intended to support learning outcomes including:

1. Multi-table data integration and querying competency through correct joins, aggregation, and transformations over normalized schemas.
2. Temporal reasoning and event-log interpretation through queries requiring time-based filtering, sequencing, and dependency reasoning.
3. Point-in-time ("as-of") analysis through reconstruction of entity state from event histories and snapshots.
4. Feature engineering from operational histories, including lag features, durations, counts, and derived KPIs.
5. Awareness and prevention of temporal leakage in model-ready dataset construction.
6. Grain consistency and avoidance of double-counting errors in multi-table joins.
7. Pipeline-oriented reasoning through ingestion, validation, and transformation of operational source data into curated analytical datasets.

1.4 Contributions

This paper makes four primary contributions. First, it articulates the limitations of static relational datasets in DSAI-oriented database and pipeline education and motivates a process-based alternative. Second, it presents the architecture and design principles of RelSim, including a three-phase pipeline that connects declarative schema configuration to simulation-driven event generation. Third, it demonstrates the approach through a healthcare case study, reports technical validation results, and provides preliminary usability feedback from graduate students comparing RelSim to commercial simulation alternatives. Finally, it outlines a planned Spring 2026 classroom deployment and evaluation framework for assessing RelSim's educational impact on student learning outcomes related to temporal reasoning, feature engineering, and reproducible data science workflows.

1.5 Paper Organization

The remainder of this paper is organized as follows. Section 2 reviews relevant background and related work in educational relational databases, synthetic relational data generation, and simulation-based approaches. Section 3 presents the design motivation for RelSim and explains why process-based causal generation is well-suited to DSAI-oriented instructional needs. Section 4 describes the RelSim system architecture and configuration framework. Section 5 presents validation results demonstrating referential integrity, temporal consistency, and coherence of the generated relational dataset. Section 6 describes the planned Spring 2026 classroom deployment and outlines an educational evaluation framework aligned with the research questions and intended learning outcomes. Finally, Section 7 concludes with a summary of contributions and directions for future work.

2. RELEVANT BACKGROUND AND RELATED WORK

Relational databases remain central to modern analytics, data engineering, and artificial intelligence (AI) systems, and database education continues to be a foundational component of many data science curricula. However, instructors face persistent challenges in sourcing datasets that are both realistic and reusable across multiple cohorts. This section reviews commonly used

educational relational datasets, summarizes major approaches to synthetic relational data generation, and situates RelSim within prior work on simulation-based synthetic data generation.

2.1 Static Relational Databases for Education

A small number of open-source relational databases have become widely used in database instruction, most notably Northwind and Sakila. These databases provide coherent schemas, realistic table relationships, and interpretable business contexts (e.g., retail ordering, inventory management, film rentals), making them valuable for teaching fundamental SQL querying and relational design concepts. Additional commonly used instructional databases include AdventureWorks, Chinook, and various small "sample business databases" distributed with relational database management systems or online tutorials.

While these datasets are pedagogically useful, they share a common limitation: they are typically static snapshots of a business domain. Once students have explored the schema and learned the "shape" of the data, they can often reuse solutions across assignments and cohorts, reducing their value as long-term instructional resources. More importantly, static datasets do not naturally support learning activities that require temporal evolution, such as incremental ETL/ELT workflows, periodic reporting cycles, or the handling of slowly changing dimensions (SCDs). In modern analytics engineering practice, analytical systems are frequently built around monthly or daily snapshots, history tables, and Type 2 dimension updates, yet educational datasets rarely contain coherent longitudinal changes that enable these concepts to be taught through realistic hands-on exercises. As a result, instructors often resort to manually modifying datasets or constructing artificial change logs, which is time-consuming and difficult to scale.

These limitations motivate the need for instructional datasets that can be regenerated with controlled variation and that reflect realistic temporal behavior rather than serving only as fixed examples.

2.2 Statistical Replication Approaches for Synthetic Relational Data

Synthetic data generation has received substantial attention in recent years due to its relevance to privacy, data sharing, and the development of reproducible analytics workflows. Many widely used approaches are based on learning statistical patterns from an existing dataset and generating new records that preserve those patterns. In this paper, we refer to this category as statistical replication, since the objective is to replicate the observed distributional properties of an existing dataset.

Early synthetic data generation techniques relied on parametric distribution fitting and rule-based generation, but more recent methods employ machine learning models capable of learning complex dependencies. Generative adversarial networks (GANs), variational autoencoders (VAEs), and diffusion-based models have been applied to tabular data generation and extended to multi-table relational settings. These approaches can capture high-dimensional correlations and generate realistic marginal distributions, making them useful for privacy-preserving data publication and synthetic training data generation.

Several software platforms operationalize these methods. One widely used open-source framework is the Synthetic Data Vault (SDV) [2], which supports the generation of synthetic tabular and relational datasets by modeling distributions and dependencies learned from real data. Such tools can produce statistically similar synthetic tables and preserve certain relational constraints, and they represent an important class of methods for generating synthetic datasets at scale.

Despite these strengths, statistical replication approaches face challenges that are particularly relevant in educational contexts. First, they typically require access to representative real data for training, which is often unavailable in instructional settings. Second, while these approaches can preserve distributions, they may struggle to generate datasets that reflect realistic temporal causality and process-driven ordering of events, especially when time is an implicit attribute distributed across multiple related tables. Third, many educational exercises require the deliberate creation of rare or targeted scenarios—such as a staffing bottleneck, a surge in demand, or a pattern of project overruns—that may not exist in the training data and may be difficult to induce through purely statistical sampling. Finally, statistical replication tools often provide limited interpretability regarding why specific records were generated or how cross-table relationships emerged, reducing their usefulness for teaching provenance and causal reasoning.

For these reasons, statistical replication approaches are valuable but may not fully address the instructional need for configurable relational datasets whose temporal behavior and provenance are explicitly controlled.

2.3 Simulation-Based and Process-Based Synthetic Data Generation

An alternative approach to synthetic data generation is to model the underlying processes that generate data. Simulation-based methods—such as discrete-event simulation and agent-based simulation—have long been used in operations research and industrial engineering to model systems involving arrivals, queues, resource constraints, and workflow dependencies. In many real organizations, relational databases serve as the operational record of such processes; therefore, simulation offers a natural mechanism for generating realistic multi-table data with coherent timestamps, waiting times, and causal ordering.

Simulation-based synthetic data generation has been explored in several domain-specific contexts. A prominent example is Synthea [], an open-source system that generates synthetic electronic health records by simulating patient lifecycles and healthcare encounters. Similar simulation-based approaches have been applied in domains such as logistics, manufacturing, and cybersecurity, where realistic process traces are needed for evaluation or training purposes. These systems demonstrate that simulation can produce highly realistic synthetic datasets that reflect operational dynamics and temporal evolution.

However, simulation-based generators are typically designed for specific domains and are often tightly coupled to particular schema assumptions and process models. As a result, they are not easily repurposed as general tools for generating synthetic relational databases across a wide

range of instructional contexts. Furthermore, many simulation-based generators focus primarily on producing event logs or traces rather than producing full normalized relational databases that are suitable for teaching relational integration and analytics pipeline development.

This gap suggests an opportunity for a general-purpose simulation-driven generator that can produce both relational databases and event traces from a configurable specification language.

2.4 Summary and Gap Addressed by RelSim

Prior work demonstrates that educational relational datasets are useful but limited by their static nature, making them difficult to reuse across cohorts and insufficient for teaching modern analytics engineering concepts such as incremental pipelines and slowly changing dimensions. Statistical replication tools such as SDV and related machine learning approaches can generate realistic synthetic tables and multi-table data when training data is available, but they may struggle to ensure temporal causality, interpretability, and controlled scenario generation in the absence of representative source datasets. Simulation-based generators demonstrate strong temporal realism and causal coherence, but existing systems are typically domain-specific and not designed as configurable tools for broad educational use.

RelSim is designed to address this gap by enabling instructors to generate synthetic relational databases and associated event traces as the outcome of explicitly defined operational workflows. By combining schema-driven entity generation with discrete-event simulation, RelSim supports temporal realism, process-driven provenance, and periodic update scenarios that are difficult to construct using static datasets or purely statistical replication methods. In doing so, RelSim provides a practical foundation for DSAI-oriented instructional activities involving multi-table integration, event-log reasoning, feature engineering from temporal histories, and the development of reproducible analytics pipelines.

3. DESIGN MOTIVATION

The design of RelSim is motivated by the growing gap between the types of datasets available for classroom use and the types of data environments students encounter in modern analytics engineering and AI workflows. Although relational database instruction has traditionally relied on static example datasets such as Sakila or Northwind, these datasets are limited in size, realism, and variability. They also fail to reflect the operational dynamics that generate real enterprise data. As a result, students often learn SQL and schema concepts in isolation, without experiencing the challenges of integrating evolving multi-table data, reconstructing point-in-time state, or building reproducible pipelines that support downstream modeling.

RelSim was designed to address this gap by enabling instructors to generate synthetic relational databases that resemble real operational systems, including both normalized relational tables and associated event traces. In contrast to many existing synthetic data tools that focus primarily on statistical similarity, RelSim adopts a process-first approach in which data is generated as the outcome of explicitly modeled workflows. This section describes the instructional and technical challenges that motivate this design.

3.1 Limitations of Static Educational Relational Datasets

Static educational databases provide useful schemas and realistic table relationships, but they impose several constraints on instructional use. First, they are difficult to reuse across cohorts because students can quickly locate solutions online or share results across semesters. Second, static datasets typically provide only a single fixed "snapshot" of the business environment and therefore do not support exercises requiring longitudinal change. Third, static datasets often contain limited or unrealistic timestamp behavior, making it difficult to design assignments involving time-based aggregation, event sequencing, or lifecycle analysis.

These limitations are particularly problematic in modern data science and AI education, where students must increasingly work with operational datasets that evolve over time and must learn to create reliable analytical datasets through iterative transformation and validation.

3.2 Importance of Temporal Evolution and Periodic Updates

Many analytics engineering workflows are fundamentally centered on the fact that source databases change over time. In real organizations, customers change addresses, employees change roles, projects change status, and transactions accumulate continuously. Analytical systems must therefore incorporate update mechanisms such as snapshot tables, incremental ingestion, and slowly changing dimensions (SCDs). Teaching these concepts requires datasets that contain realistic entity evolution and periodic update cycles (e.g., monthly snapshots).

Constructing such evolving relational datasets is difficult using static sources. While an instructor can manually modify a dataset to simulate change, doing so repeatedly across multiple cohorts is time-consuming and often produces artificial changes that do not reflect realistic business processes. In contrast, if entity updates emerge naturally from simulated workflows, periodic database snapshots can be generated coherently, enabling students to work with multi-period datasets that resemble the operational-to-analytical pipeline challenges encountered in practice.

Supporting realistic periodic updates and SCD-driven workflows was therefore a central motivation for RelSim's simulation-based design.

3.3 Why Relational Synthesis is Harder than Single-Table Synthesis

Synthetic generation of single-table datasets is relatively straightforward: values can be sampled from specified distributions or generated using deterministic rules. However, generating coherent multi-table relational data is significantly more challenging because correctness is defined not only by individual column distributions but also by cross-table consistency and process realism. RelSim's design is motivated by four core technical challenges.

First, relational databases must satisfy structural and referential integrity constraints. Foreign key relationships must remain valid, primary keys must remain unique, and relationship cardinalities

(one-to-many, many-to-many) must be internally consistent. Even minor violations can render a dataset unusable for realistic instruction.

Second, relational datasets contain important cross-table statistical dependencies that emerge only when tables are joined. For example, customer value distributions, project profitability, consultant utilization, or resource workloads are often not represented in a single table but are derived through multi-table integration. Preserving these dependencies—sometimes referred to as join fidelity—is essential for generating data that supports realistic analytical queries and modeling exercises.

Third, realistic relational databases are shaped by the temporal and causal ordering of events. In operational systems, records are not created independently; they reflect sequences of actions such as requests, approvals, assignments, completions, and billing. Time-based fields therefore cannot be generated arbitrarily without violating causality (e.g., a payment cannot occur before an invoice is issued). These dependencies become more complex when multiple entities interact through many-to-many relationships and when processes include waiting, resource contention, and branching workflows.

Fourth, realistic databases encode implicit business rules and semantics that are not fully captured in schema constraints. Examples include limits on staffing capacity, client contract rules, project milestone dependencies, or eligibility conditions for specific actions. In real systems, these rules are often enforced through application logic or workflow constraints rather than through database constraints alone. As a result, simply generating statistically plausible values does not guarantee that the dataset reflects a realistic operational environment.

These challenges collectively motivate the need for a generative approach that can enforce relational integrity and causal ordering by construction rather than through post-hoc correction.

3.4 Statistical Replication versus Causal Generation

Existing synthetic relational data generators often rely on learning patterns from an existing database and generating new records that replicate observed distributions. As discussed in Section 2, these statistical replication methods are valuable in privacy and data-sharing contexts, but they may be less suitable for educational use when representative training data is unavailable or when instructors wish to deliberately generate scenario-driven datasets.

RelSim instead adopts a causal generation approach in which data is produced as the outcome of simulated operational workflows. In this approach, the process model defines how entities are created, how they evolve over time, and how relationships between entities arise through events such as assignments, service encounters, approvals, or billing cycles. Rather than independently sampling timestamps or relational links, RelSim uses discrete-event simulation to produce time-stamped records that reflect waiting, capacity constraints, and causal dependencies. This approach yields datasets whose temporal structure and provenance are interpretable and configurable, enabling instructors to generate multiple dataset variants while preserving realism.

3.5 Design Objectives

Based on these instructional and technical motivations, RelSim was designed to meet the following objectives:

- Generate coherent normalized relational databases that satisfy referential integrity and realistic cardinalities.
- Produce realistic temporal histories by simulating event sequences rather than sampling timestamps independently.
- Support periodic snapshot generation to enable instruction involving incremental pipelines and slowly changing dimensions.
- Provide instructor-configurable variability, allowing datasets to be regenerated across cohorts with controlled differences.
- Enable event traces alongside relational tables, supporting teaching activities involving provenance, temporal joins, and feature engineering.
- Remain domain-flexible, allowing instructors to define new schemas and workflows without rewriting the system.

These objectives guided the design of RelSim's three-phase architecture, described in the next section.

4. SYSTEM OVERVIEW

RelSim is an open-source platform designed to generate synthetic relational databases and associated event traces for instructional use in data science, analytics engineering, and AI curricula. Unlike traditional synthetic data generators that attempt to replicate statistical patterns from an existing dataset, RelSim produces relational datasets as the outcome of explicitly modeled operational workflows. This simulation-driven approach enables the generation of coherent multi-table datasets with realistic temporal behavior, causal ordering, and cross-table provenance.

RelSim is organized as a three-phase pipeline (Figure 1): Entity Generation, Discrete-Event Simulation, and Post-Simulation Derivation. Together, these phases enable instructors to define a domain scenario, generate an initial population of entities, simulate operational activity over time, and materialize a normalized relational database suitable for SQL, pipeline, and feature engineering assignments

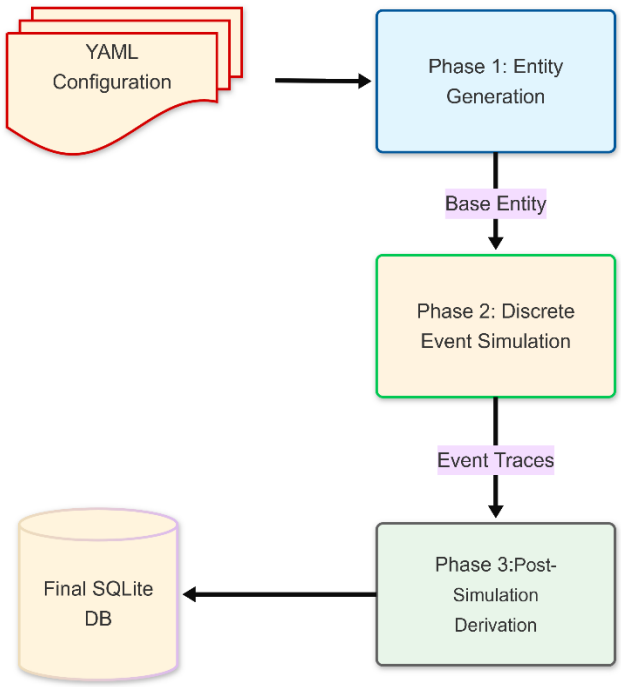

Figure 1: System Overview

4.1 Phase 1: Static Entity Generation

In Phase 1, RelSim generates the initial population of entities defined by the schema configuration. This includes the creation of key dimension-like tables and reference entities such as customers, patients, consultants, facilities, or services depending on the domain. Attribute values are generated using instructor-defined templates, formulas, and probability distributions.

The primary goal of this phase is to establish a coherent baseline database state and ensure that primary key and foreign key structures are valid prior to simulation. By generating entity populations separately from the simulation logic, RelSim allows instructors to vary dataset scale and composition without modifying the underlying process model.

4.2 Phase 2: Discrete-Event Simulation

In Phase 2, RelSim executes a discrete-event simulation of the operational workflows defined in the configuration. Events are generated according to arrival processes and activity logic, and may include waiting times, branching decisions, and resource constraints such as staffing availability or capacity limits. Each event is time-stamped and recorded in an event trace, enabling realistic temporal sequencing and causal ordering.

This simulation-driven approach is central to RelSim's design. Rather than independently sampling timestamps for records, RelSim generates time-based behavior through explicit process logic. As a result, the resulting database reflects realistic operational patterns such as queues,

delays, workload variability, and dependent relationships across entities. This capability supports instructional exercises involving temporal joins, point-in-time analysis, and event-driven feature engineering.

4.3 Phase 3: Post-Simulation Derivation

In Phase 3, RelSim transforms the raw simulation trace into relational tables suitable for instructional use. This post-processing step materializes tables whose records depend on multiple simulated events and relationships, including many-to-many relationship tables, history tables, and periodic snapshot tables.

Post-Simulation Derivation is particularly important for ensuring temporal consistency across complex relationships. For example, many-to-many relationships often require intermediate logic to determine when a relationship becomes valid, how long it persists, and how it should be represented in a normalized schema. Similarly, RelSim can generate periodic database snapshots (e.g., monthly states), enabling instructional activities involving incremental pipelines and slowly changing dimensions.

The output of this phase is a complete normalized relational database instance, accompanied by event trace data that preserves operational provenance.

4.3 Output Artifacts and Instructional Relevance

RelSim produces two primary output artifacts: (1) a relational database instance conforming to the configured schema, and (2) an event trace capturing time-stamped operational activity. These outputs support a broad range of instructional use cases, including multi-table SQL querying, data integration and transformation workflows, temporal feature engineering, and pipeline orchestration exercises. Additionally, because RelSim can generate coherent periodic updates and snapshots, it provides a foundation for teaching incremental ingestion patterns and slowly changing dimensions within analytics engineering curricula.

The next section presents validation results demonstrating the internal consistency of the generated database, including referential integrity and temporal coherence across simulated events and derived relational tables.

5. RESULT VALIDATIONS AND CONSISTENCY CHECKS

A primary design goal of RelSim is to generate synthetic relational databases that are not only statistically plausible but also structurally coherent and temporally consistent. Because RelSim produces multi-table datasets through simulation-driven workflows, validation must ensure that generated records satisfy relational integrity constraints and that timestamps and derived relationships remain consistent with causal ordering. This section presents a set of validation checks applied to the healthcare case study dataset to confirm that the generated database is internally coherent and suitable for instructional use.

It is important to emphasize that the results in this section represent technical system validation, not educational evaluation. These checks demonstrate that RelSim produces databases that satisfy integrity and temporal coherence requirements, but they do not yet measure student learning outcomes or instructional effectiveness. Educational evaluation is addressed in Section 6 through a planned deployment and assessment framework.

5.1 Healthcare Case Study Context

To validate RelSim's ability to generate coherent multi-table datasets, we applied the system to a healthcare scheduling and service delivery scenario. This case study was selected because it contains realistic operational complexity, including multiple interacting entity types (patients, providers, facilities, and appointments), time-dependent workflows, and many-to-many relationships that evolve over time. The resulting dataset includes both normalized relational tables and an event trace capturing time-stamped operational activity.

Figure 2 presents the relational schema used in this case study, illustrating the core entities and their foreign key relationships. Figure 3 summarizes the simulation workflow used to generate operational activity over time, including event sequencing, resource constraints, and dependent process logic. Together, these figures provide the structural and process context for the validation results presented in the remainder of this section.

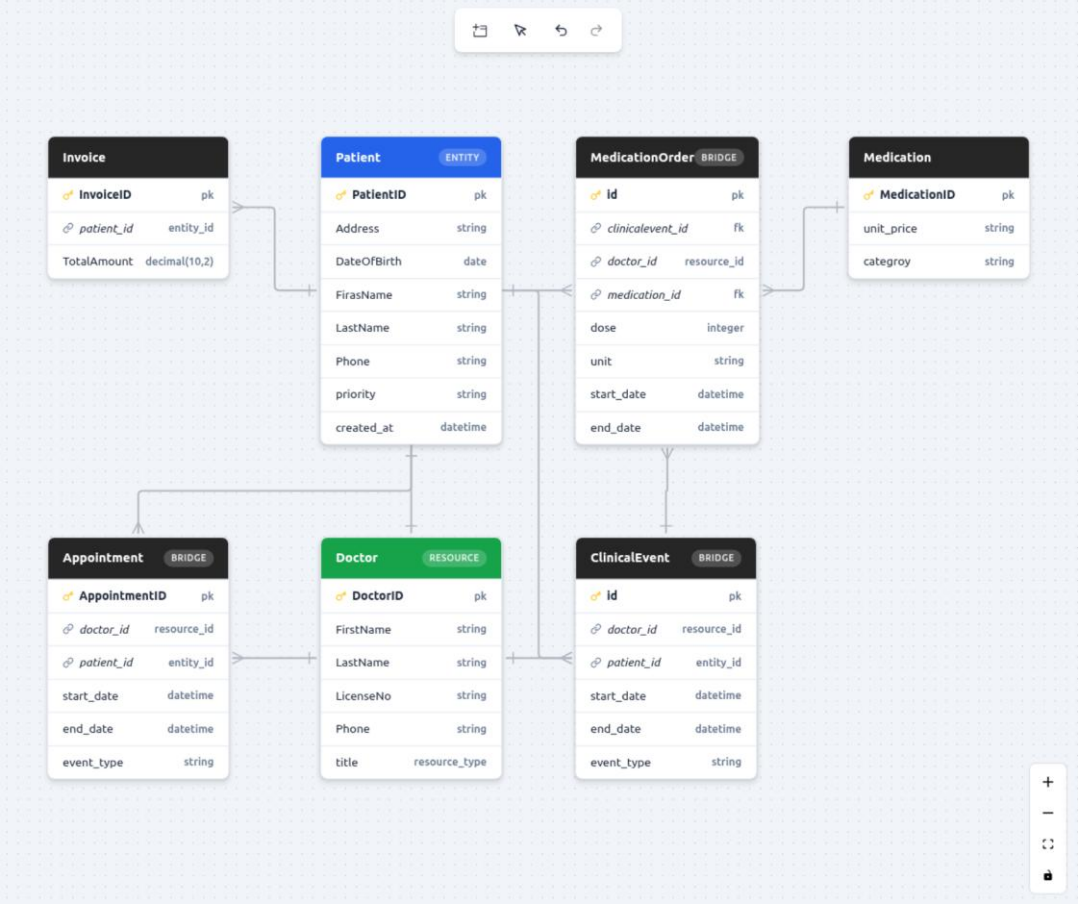

Figure 2: Healthcare Schema in RelSim Schema Designer

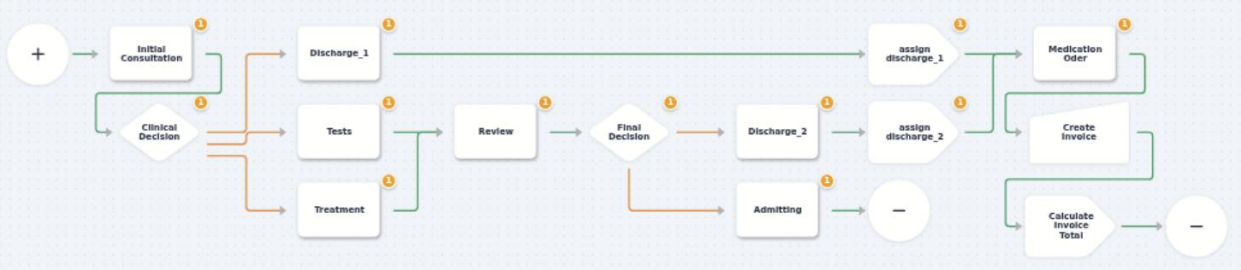

Figure 3: Healthcare Process Model in RelSim Simulation Designer

5.2 Referential Integrity and Structural Consistency

The most fundamental requirement for a synthetic relational database is that it satisfies basic structural correctness. RelSim enforces primary key uniqueness during entity generation and ensures that foreign key references remain valid during simulation and post-simulation derivation. For the healthcare case study, all generated tables were validated to confirm that foreign key values correspond to existing primary keys and that no orphan records were produced.

In addition to foreign key checks, table-level cardinality and relationship constraints were validated. For example, patient-to-appointment and provider-to-appointment relationships must follow expected one-to-many patterns, while relationships such as provider assignment or patient encounters may involve many-to-many mappings over time. These checks confirm that the generated dataset maintains consistent relational structure and is suitable for multi-table SQL querying and data integration exercises.

5.3 Temporal Consistency and Causal Ordering

Beyond referential integrity, realistic instructional datasets must exhibit coherent temporal behavior. RelSim's simulation-driven approach generates timestamps as the outcome of operational workflows, which enables causal ordering constraints to be satisfied by construction. However, validation is still necessary to confirm that post-simulation derivations preserve these constraints across tables.

For the healthcare case study, temporal consistency checks were applied to ensure that event timestamps reflect valid lifecycle orderings. Examples include ensuring that appointment records occur after patient creation, that service events occur only after appointment scheduling, and that dependent records do not appear before prerequisite actions. These checks are essential for supporting instructional exercises involving temporal joins, window functions, and point-in-time reconstruction of entity state.

The results indicate that RelSim successfully generates temporally coherent event sequences across related tables. This coherence is particularly important for analytics engineering and AI workflows, where students must often build features from event histories and must avoid temporal leakage when constructing modeling datasets.

5.4 Many-to-Many Temporal Integrity

Many-to-many relationships are a common source of inconsistency in synthetic relational generation, particularly when relationships evolve over time. In operational systems, many-to-many mappings are often not static associations but are established through time-dependent events (e.g., assignment, reassignment, completion). RelSim addresses this challenge through its Phase 3 Post-Simulation Derivation, which materializes many-to-many relationship tables based on the simulation trace.

To validate this approach, additional checks were applied to ensure that derived many-to-many records correspond to valid simulated relationships and remain temporally consistent. For example, assignments between patients and providers must only be active during appropriate time intervals, and derived relationship tables must reflect these time-bounded associations. These validations demonstrate that RelSim can generate complex relational structures that are difficult to reproduce using static generation or purely statistical replication.

5.5 Entity Lifecycle Realism: Patient Journey Timelines

While referential and temporal checks confirm internal consistency, it is also important to validate that the generated data reflects realistic entity lifecycles. Figure 10 provides a visualization of patient journey timelines, illustrating how individual patients progress through the simulated healthcare workflow over time. This figure highlights the sequential and event-driven nature of the generated dataset, including realistic waiting periods, repeated encounters, and workflow-driven transitions.

The patient journey visualization provides qualitative evidence that RelSim generates coherent temporal histories at the entity level rather than producing isolated records with independently sampled timestamps. Such lifecycle realism is important for instructional scenarios requiring students to interpret longitudinal histories, reconstruct state transitions, or engineer features based on past activity.

5.6 Cross-Table Provenance and Data Lineage

A key motivation for RelSim is to generate datasets with interpretable provenance: records should be traceable to the events and processes that produced them. Figure 11 illustrates cross-table data lineage, showing how simulated events propagate into multiple relational tables and how derived records can be traced back to operational activity.

This lineage perspective is particularly valuable in DSAI-oriented education because it supports learning objectives related to causal reasoning, reproducibility, and pipeline development.

Students working with RelSim-generated datasets can examine not only “what the data contains” but also “how the data came to exist,” enabling instructional exercises involving auditability, event-to-table mapping, and the construction of analytical datasets from operational sources.

5.7 Preliminary Usability Feedback

While full classroom deployment is scheduled for Spring 2026, a preliminary usability pilot was conducted with three graduate research assistants who assisted in developing the healthcare case study configuration. The students were tasked with translating the high-level healthcare workflow into RelSim’s declarative YAML specifications and "Database Canvas".

Feedback from this pilot was highly positive regarding the system's "causal generation" approach. Specifically, all three participants noted that specifying complex process models in RelSim was significantly more straightforward and intuitive than using the commercial discrete-event simulation (DES) packages widely used in industry. They highlighted that because RelSim is purpose-built for database generation, it removes the overhead of modeling physical movement or complex visual animations required by general-purpose DES tools, allowing them to focus strictly on the logical events and state transitions that generate relational records. This suggests that RelSim may lower the barrier for instructors to create custom domain-specific datasets compared to existing industrial simulation solutions.

5.8 Summary of Validation Results

Across the healthcare case study, RelSim successfully generated a normalized multi-table relational database satisfying key correctness criteria. Referential integrity checks confirmed that foreign key relationships were valid across all tables. Temporal consistency checks verified that event ordering and lifecycle constraints were respected. Additional validations confirmed that many-to-many relationships derived in Phase 3 remained temporally consistent and traceable to simulation events. Qualitative inspection of entity timelines (Figure 4) and cross-table lineage structure (Figure 5) further demonstrated that RelSim produces realistic temporal histories and coherent provenance.

In addition to these technical checks, qualitative feedback from student developers indicates that RelSim’s process-oriented configuration is more accessible for data-centric tasks than traditional industrial simulation software, supporting its viability as a tool for curriculum development.

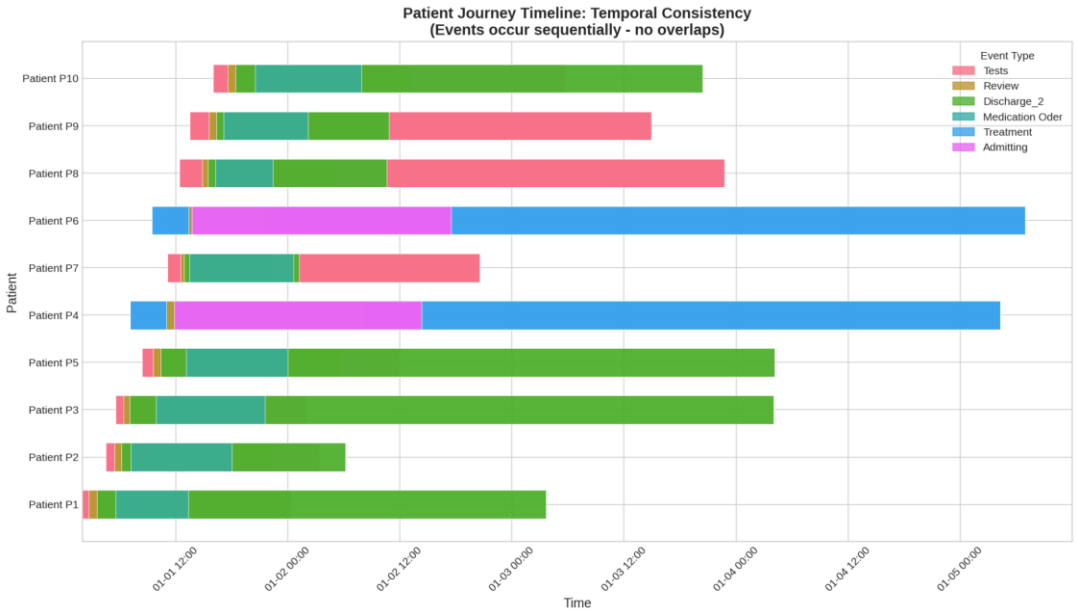

Figure 4: Entity Timelines

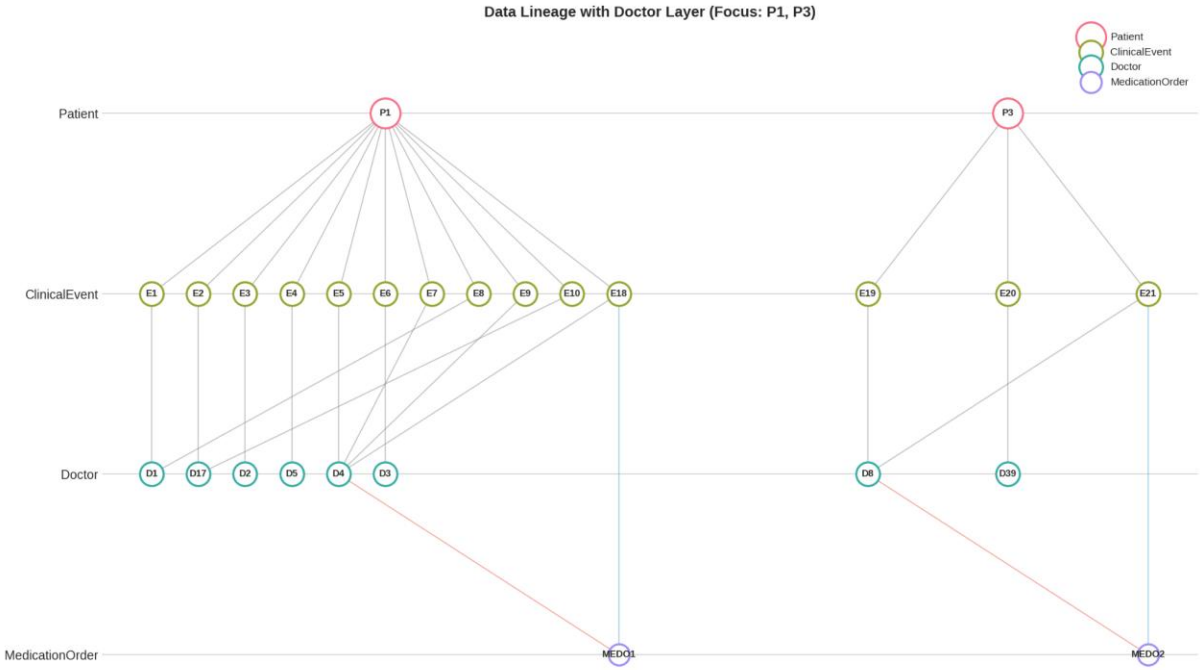

Figure 5: Lineage Structure

Collectively, these results demonstrate that RelSim produces internally consistent relational databases and event traces suitable for use as instructional datasets in data science, analytics engineering, and AI curricula. The next section describes the planned Spring 2026 classroom

deployment and outlines an evaluation framework designed to assess educational outcomes associated with using RelSim-generated datasets in a project-based learning setting.

6. PRELIMINARY CLASSROOM USE & PLANNED EDUCATIONAL DEPLOYMENT

RelSim was developed to address a practical instructional challenge: the limited availability of reusable, realistic relational datasets that support modern analytics engineering and AI workflows. While the validation results in Section 5 demonstrate that RelSim produces internally coherent relational databases with realistic temporal behavior, the primary goal of this work is educational. This section describes the planned Spring 2026 classroom deployment of RelSim and presents an evaluation framework designed to assess its instructional value.

Because the system has not yet been deployed in a formal classroom setting at the time of submission, the results presented in this section are forward-looking and are intended to provide a learning-aligned evaluation plan that can be executed during the planned deployment.

6.1 Course Context and Intended Use

RelSim will be deployed in Spring 2026 in a master's-level course in analytics engineering. Students in this course typically have prior exposure to SQL and basic analytics workflows but are still developing proficiency in integrating operational data sources, building reproducible pipelines, and producing curated analytical datasets suitable for reporting and machine learning.

The primary instructional use of RelSim will be to generate a realistic normalized operational database for a hypothetical consulting firm. The generated database includes 13 interrelated tables representing clients, projects, deliverables, consultants, staffing assignments, billing activity, and related operational events. The dataset will serve as the "source system" for a course project in which student teams build an orchestrated data pipeline that ingests operational data, validates it, transforms it into curated analytical tables, and produces reporting-ready outputs.

This deployment is intended to support a more authentic pipeline development experience than is possible with static instructional databases, particularly by enabling time-based updates, periodic snapshots, and scenario-driven variations.

6.2 Instructional Scaffolding and Student Deliverables

RelSim will be incorporated into the course through a structured sequence of instructional activities. Students will be introduced to the generated consulting-firm database early in the term and will be provided with a data dictionary and schema diagram to support onboarding. The project will then proceed through a staged pipeline development process consistent with modern analytics engineering practice.

Planned student deliverables include:

1. **Source system exploration and profiling**, including schema interpretation and basic data quality checks.
2. **Multi-table integration queries**, including joins across normalized tables and the creation of derived measures such as utilization, project cost accumulation, and revenue recognition.
3. **Construction of curated analytical datasets**, including the creation of fact-style tables and dimension-style reference tables appropriate for reporting and dashboarding.
4. **Temporal aggregation and snapshot reporting**, including the creation of periodic monthly project status summaries and historical tables suitable for slowly changing dimension handling.
5. **Pipeline automation and orchestration**, in which student teams implement a reproducible workflow that executes ingestion, transformation, and validation tasks on a recurring schedule.
6. **Optional feature engineering extension**, where students construct a modeling dataset (e.g., predicting project overruns or client churn) using temporal history and operational event traces.

These deliverables are designed to emphasize core DSAI-relevant competencies including relational reasoning, temporal joins, feature construction from operational history, and prevention of temporal leakage when creating model-ready datasets.

6.3 Intended Learning Outcomes

RelSim is intended to support learning outcomes that are difficult to achieve using static educational databases. In particular, the planned deployment targets the following outcomes:

- **Multi-table integration competency**: Students will be able to correctly integrate normalized operational tables through joins and aggregations while maintaining appropriate grain.
- **Temporal reasoning and lifecycle analysis**: Students will be able to reason about event ordering, reconstruct entity state over time, and interpret process-driven event histories.
- **Pipeline-oriented thinking**: Students will be able to implement repeatable ingestion and transformation workflows that mirror real analytics engineering practice.
- **SCD and periodic update reasoning**: Students will be able to construct snapshot tables and manage historical dimension changes consistent with slowly changing dimension concepts.
- **Feature engineering from operational history**: Students will be able to derive meaningful features from event traces and time-dependent tables.
- **Leakage awareness**: Students will be able to identify and avoid temporal leakage when constructing modeling datasets.

6.4 Evaluation Plan and Data Collection Strategy

To address the need for educational evidence identified by reviewers, the Spring 2026 deployment will include a structured evaluation plan aligned with the research questions defined in Section 1.2. The evaluation will incorporate both quantitative performance measures and qualitative student feedback.

6.4.1 Assessment Instruments

Planned evaluation instruments include:

- **Pre- and post-assessment quiz** focused on temporal reasoning, multi-table query logic, grain consistency, and SCD concepts. Items will include short SQL interpretation questions and conceptual reasoning prompts.
- **Project rubric-based scoring**, evaluating team deliverables including correctness of curated datasets, quality of transformations, documentation quality, and temporal validity of snapshot tables.
- **Artifact analysis**, including review of submitted SQL scripts, transformation code, and pipeline logs to identify common error patterns (e.g., double-counting, incorrect joins, leakage).
- **Student perception survey**, capturing perceived realism, engagement, and instructional usefulness of the dataset relative to prior experiences with static educational databases.
- **Instructor observation and reflection**, documenting the time required to generate dataset variants, ease of integration into course assignments, and perceived instructional value.

6.4.2 Comparative Baseline

To provide a meaningful comparison point, outcomes from the Spring 2026 deployment will be compared to prior course offerings that relied on static datasets and manually constructed assignment data. Although the course structure will not be identical across semesters, comparison will focus on specific deliverables and skills such as multi-table integration performance, temporal reasoning quiz results, and observed error rates in pipeline submissions.

6.4.3 Research Questions and Evaluation Alignment

The planned evaluation is structured around the following research questions:

- **RQ1 (Temporal reasoning):** measured through quiz items requiring event sequencing, temporal joins, and point-in-time reasoning.
- **RQ2 (Multi-table integration performance):** measured through rubric scores and artifact analysis of SQL transformations.
- **RQ3 (Leakage prevention and modeling readiness):** measured through evaluation of feature engineering artifacts and project deliverables requiring time-aligned dataset construction.
- **RQ4 (Reusability across cohorts):** evaluated through instructor effort metrics and the ability to regenerate scenario variants with controlled differences.
- **RQ5 (Student perceptions):** measured through survey items and qualitative feedback.

6.5 Summary

RelSim's planned Spring 2026 deployment is designed to evaluate whether simulation-generated relational datasets can support more authentic and effective instruction in analytics engineering and DSAI workflows. The deployment will leverage a consulting firm database generated by RelSim as the primary operational source system for a project-based pipeline development experience. The evaluation plan includes learning-aligned assessments, artifact-based performance measures, and student feedback instruments intended to provide evidence of educational value in future iterations of this work.

7 PRELIMINARY CLASSROOM USE & PLANNED EDUCATIONAL DEPLOYMENT

This paper presented RelSim, an open-source simulation-driven platform for generating synthetic relational databases and event traces for use in data science, analytics engineering, and AI education. RelSim was motivated by a persistent instructional gap: widely used open-source educational databases such as Sakila and Northwind provide coherent schemas but are static, limited in variability, and insufficient for teaching modern workflow-oriented concepts such as temporal reasoning, provenance, periodic updates, and slowly changing dimensions. RelSim addresses this limitation through a three-phase pipeline consisting of Entity Generation, Discrete-Event Simulation, and Post-Simulation Derivation, enabling datasets to be produced as the outcome of explicit operational workflows rather than through independent sampling or purely statistical replication.

Using a healthcare scheduling case study, this paper demonstrated that RelSim can generate normalized multi-table relational datasets with strong internal coherence. Validation results confirmed referential integrity, temporal consistency, and the ability to materialize complex relationships and time-dependent structures from simulation output. Visualizations of entity lifecycle timelines and cross-table data lineage further illustrated that RelSim produces interpretable temporal histories and provenance that are difficult to obtain using static datasets or replication-based synthetic data generators.

While these results provide evidence of technical feasibility and internal consistency, the primary contribution of RelSim is educational, and its instructional impact has not yet been empirically measured. A planned Spring 2026 classroom deployment in a master's-level analytics engineering course will use a RelSim-generated consulting firm database as the primary source system for an orchestrated data pipeline project. The evaluation plan described in this paper outlines research questions, learning outcomes, and assessment instruments intended to measure whether process-generated relational datasets improve student learning in areas such as multi-table integration, temporal reasoning, SCD handling, and feature engineering from operational histories.

Future work will focus on three directions. First, the Spring 2026 deployment will provide an opportunity to collect student performance data and qualitative feedback to evaluate RelSim's educational effectiveness. Second, the system will be expanded to support additional domain configurations and richer scenario generation capabilities, enabling instructors to generate

targeted business conditions and controlled dataset variants across cohorts. Third, longer-term research will explore quantitative benchmarking against statistical replication approaches and investigate methods for calibrating simulation parameters from real datasets, potentially enabling hybrid workflows that combine distributional estimation with process-based causal generation.

RelSim represents a step toward making realistic, evolving relational datasets more accessible for instruction in modern data science and AI workflows. By generating coherent databases and event traces grounded in operational processes, RelSim provides a flexible foundation for teaching relational reasoning, temporal analysis, and pipeline-oriented analytics engineering skills in a manner that more closely reflects real-world practice.

References

[1] Hudovernik, V., Jurkovič, M., & Štrumbelj, E. (2024). Benchmarking the Fidelity and Utility of Synthetic Relational Data. arXiv:2410.03411.

[2] Walonoski, J., et al. (2017). *Synthea: An approach, method, and software mechanism for generating synthetic electronic health records*. https://synthetichealth.github.io/synthea/

[3] DataCebo. (2026). *The Synthetic Data Vault (SDV) Documentation*. https://docs.sdv.dev/sdv