SYNTHERELA: A BENCHMARK FOR SYNTHETIC RELATIONAL DATABASE GENERATION

Martin Jurkovič*, **Valter Hudovernik***, **Erik Štrumbelj**
Faculty of Computer and Information Science, University of Ljubljana

ABSTRACT

Synthesizing relational databases has started to receive more attention from researchers, practitioners, and industry. The task is more difficult than synthesizing a single table due to the added complexity of relationships between tables. For the same reason, benchmarking methods for synthesizing relational databases introduces new challenges. Our work is motivated by a lack of an empirical evaluation of state-of-the-art methods and by gaps in the understanding of how such an evaluation should be done. We review related work on relational database synthesis, common benchmarking datasets, and approaches to measuring the fidelity and utility of synthetic data. We combine the best practices, a novel robust detection metric and relational deep learning utility, a novel approach to evaluating utility with graph neural networks, into a benchmarking tool. We use it to compare 6 open source methods over 8 real-world databases, with a total of 39 tables. The open-source SyntheRela benchmark is available on GitHub, alongside a public leaderboard.
Data & Code: `github.com/martinjurkovic/syntherela`
Leaderboard: `huggingface.co/spaces/SyntheRela/leaderboard`

1 INTRODUCTION

Synthesizing relational databases - generating relational databases that preserve the characteristics of the original databases - is an emerging field. It promises several benefits, from protecting privacy to addressing data scarcity, while preserving the complexity and dependencies present in the original databases. This makes it attractive for healthcare (Appenzeller et al., 2022), finance (Assefa et al., 2020), and education (Bonnéry et al., 2019), where accessing and utilizing data can be challenging due to privacy concerns, data scarcity, or biases (Ntoutsi et al., 2020; Rajpurkar et al., 2022).

The foundations of synthesizing relational databases were laid by the Synthetic Data Vault (Patki et al., 2016). Recently several deep learning methods have been proposed (Gueye et al., 2023; Li & Tay, 2023; Mami et al., 2022; Xu et al., 2023; Canale et al., 2022; Solatorio & Dupriez, 2023; Pang et al., 2024; Hudovernik, 2024). The field has also received attention from the industry, with several commercial tools now available and with Google, Amazon, and Microsoft integrating them into their cloud services (Gretel.ai, 2024).

While there are several packages for evaluating the quality of synthetic tabular data, only the SD-Metrics package (Patki et al., 2016) provides some support for the evaluation of synthetic relational databases. As such, the field lacks not only an empirical comparison of available methods but also an understanding of how such an evaluation should be done. We address this gap with an evaluation methodology that combines established evaluation metrics (Section 2.2), best practices, sampling procedures, and real-world relational databases (Section B.2.1). We also propose a detection-based metric **C2ST-Agg** specialized for relational databases and propose a novel approach to evaluating the utility of synthetic relational databases with **relational deep learning utility**.

We implement the methodology in **SyntheRela**, a benchmark and evaluation tool that is available as an open source package and can be easily extended with new metrics and datasets (see Appendix B). Finally, we use the benchmark to evaluate current state-of-the-art methods (Section 2.1) over several

*Equal contribution.

relational databases (Section B.2.1). This is the first comprehensive evaluation and comparison of methods for the synthesis of relational databases and provides valuable insights into their ability to synthesize relational aspects of the data (Section 4). The code is publicly available alongside a public leaderboard.

2 Related Work

2.1 Methods for synthesizing relational databases

In this work, we focus on relational databases — a collection of tables linked by primary and foreign keys. We distinguish this from synthesizing tabular data (a single table), which is a special case and an even more active field (Borisov et al., 2022; Hansen et al., 2023; Qian et al., 2023b). Here we briefly summarize the methods. A detailed description can be found in Appendix A.

The Synthetic Data Vault (SDV) uses Gaussian copulas and predefined distributions. Row Conditional-TGAN (RC-TGAN) (Gueye et al., 2023) and Incremental Relational Generator (IRG) (Li & Tay, 2023) are based on GANs. The Realistic Relational and Tabular Transformer (REaLTabFormer) (Solatorio & Dupriez, 2023) and Composite Generative Models (Canale et al., 2022) are based on transformers. The work of Mami et al. (2022) is based on graph variational autoencoders, while Xu et al. (2023) propose a framework for synthesizing many-to-many datasets using random graphs. Pang et al. (2024) propose ClavaDDPM, a method based on classifier-guided diffusion models. Hudovernik (2024) proposes RGCLD, a method based on graph neural networks (GNNs) and conditional diffusion models. Recently, Tiwald et al. (2025) propose TabularARGN - a framework for tabular and relational data synthesis based on autoregressive tabular models.

2.2 Metrics for Evaluating Synthetic Data

The two main aspects for evaluating the quality of synthetic tabular data and relational databases are *fidelity* and *utility*. Fidelity measures the degree of similarity between synthetic and real data in terms of its properties, whereas utility measures how well the synthetic data can replace real data when the data are part of some tasks, for example, for predictive modeling (Hansen et al., 2023).

We further divide fidelity metrics into *statistical*, *distance-based*, and *detection-based* metrics. Utility of synthetic data is typically assessed with train-on-synthetic evaluate-on-real methods (Beaulieu-Jones et al., 2019).

Another dimension of evaluation metrics for relational data is granularity. The most common are *single-column* metrics that evaluate the marginal distributions, *two-column* metrics that evaluate bivariate distributions[1], *single-table* metrics that evaluate tables, and *multi-table* metrics that evaluate the relational aspects.

Statistical fidelity methods are typically used to assess marginal distributions, sometimes bivariate distributions. The most commonly used methods are the Kolmogorov-Smirnov test and the $\mathcal{X}^2$ test for numerical and categorical variables, respectively. For relational data, cardinality shape similarity is used, where for each parent row the number of child rows is calculated. This yields a numerical distribution for both real and synthetic data, on which a Kolmogorov-Smirnov test is performed.

Similar to statistical fidelity, **distance-based fidelity** is typically used to assess the quality of marginal distributions. However, some distance metrics also assess entire tables. Commonly used distance-based methods are total variation distance, Kullback-Leibler divergence, Jensen-Shannon distance, Wasserstein distance, maximum mean discrepancy, and pairwise correlation difference. To evaluate inter-table relationships Pang et al. (2024) use k-hop similarity, where they compute correlations between tables at distance k [2]. Unlike statistical methods, reports of distance-based fidelity do not include hypothesis testing or any other quantification of uncertainty. This is an issue both when evaluating a method and when comparing two methods. In the former, a method can achieve a seemingly high distance that is in a high probability region when taking into account the sampling

[1] In this work we group two-column (bivariate) metrics under single-table metrics.

[2] e.g., 0-hop refers to columns within the same table, while 1-hop refers to columns in tables directly connected via a foreign key.

distribution. In the latter, a seemingly large difference between the two methods can be explained away by the variance of the sampling distribution.

The basic idea of **detection-based fidelity** is to learn a model that can discriminate between real and synthetic data. The detection-based metric can be interpreted as a null-hypothesis test for comparing two distributions (two sample testing) with classification accuracy as a proxy Kim et al. (2021). The classifier serves as a map from high-dimensional data to a one-dimensional test statistic. In machine learning literature, this is referred to as a classifier two sample test (C2ST) (Lopez-Paz & Oquab, 2017). If the model can achieve better-than-random predictive performance, this indicates that there are some patterns that identify synthetic data. Zein & Urvoy (2022) show that using discriminative models can highlight the differences between real and synthetic tabular data.

The most common detection-based metric is logistic detection (LD) (Gueye et al., 2023; Solatorio & Dupriez, 2023; Li & Tay, 2023; Pang et al., 2024), where a logistic regression model is used for discrimination. An extended version of LD known as parent-child logistic detection (P-C LD) is used to evaluate relational databases. P-C LD applies LD to denormalized pairs of synthetic parent and child tables, assessing the preservation of parent-child relationships. A serious issue with denormalization is that it may introduce correlation between rows, breaking the iid assumption. This results in an over-performance of the discrimintative model and in underestimating the quality of the method for synthesizing relational data. It also makes it impossible to set a detection threshold for testing fidelity (for example, accuracy would be greater than 50% even if both datasets were from the same data generating process). For these reasons, we do not consider P-C detection.

Note that logistic regression is unable to capture interactions between columns unless these interactions are explicitly included as features. This implies a lenient evaluation of the state-of-the-art methods (we demonstrate this empirically in Appendix C.3). Tree-based ensemble models are a better alternative, which is also suggested by the findings of Zein & Urvoy (2022) for tabular data.

The **utility** of synthetic data is most commonly measured with machine learning efficacy (ML-E) - comparing the hold-out performance of a predictive model trained on the original data with a predictive model trained on synthetic data (Canale et al., 2022; Li & Tay, 2023; Mami et al., 2022; Solatorio & Dupriez, 2023; Pang et al., 2024). Patki et al. (2016) measured utility with a user study and Hansen et al. (2023) with the ability to retain model or feature importance ranking (measured with rank correlation) in the train-on-synthetic evaluate-on-real paradigm. Note that all of these studies evaluated utility on a single table, even those that investigated synthetic relational databases.

3 Evaluating Synthetic Relational Databases

3.1 Multi-table Fidelity Using Aggregation

We first address the lenient evaluation by replacing logistic regression commonly used in related work with tree-based methods (Borisov et al., 2023; Zein & Urvoy, 2022).

Our next observation is that current fidelity metrics fail to thoroughly evaluate relationships between tables: (i) cardinality similarity assesses only the cardinality of foreign key relationships, (ii) k-hop similarity evaluates only linear relationships between columns in related tables, and (iii) denormalization breaks the iid assumption required for C2ST, leading to unreliable fidelity assessments.

To address these limitations, we introduce a classifier two-sample test with aggregation (C2ST-Agg). Instead of denormalizing tables, we preserve the multi-table structure by augmenting both real and synthetic parent tables with aggregated features derived from their child tables. Aggregation is an established technique in the field of relational reasoning (Getoor et al., 2007; Džeroski, 2010) and C2ST-Agg can be thought of as a propositionalization (Kramer et al., 2001) approach to the C2ST on relational databases. By summarizing child-table columns and relationship cardinality through aggregation functions (e.g., mean, count, max), C2ST-Agg maintains the iid assumption while enhancing classifier-based fidelity assessment. Our approach addresses the issues of current fidelity metrics: it accounts both for relationship cardinality (i) and high-level interactions across all columns in related tables (ii), while maintaining the iid assumption for each table (iii). In practice, users can select aggregation functions based on the specific aspects of relational data they wish to evaluate. We provide general guidelines for choosing these aggregations, along with a detailed explanation of the C2ST-Agg metric, in Appendix B.1.1

3.2 RELATIONAL DEEP LEARNING UTILITY

Recently, relational deep learning has emerged as an alternative to traditional ML methods by transforming relational databases into graphs (Fey et al., 2023; Papamarkou et al., 2024) and utilizing graph neural networks (GNNs). Subsequently, Robinson et al. (2024) proposed RelBench, a benchmark for relational deep learning. The authors of RelBench show that the performance of a GNN pipeline is comparable to a traditional ML pipeline approach done by a data scientist.

To omit the creation of feature engineering pipelines to transform relational databases into a single table and then train ML models, we incorporate the graph approach of RelBench into SyntheRela. We use the RelBench GNN pipeline to fit GraphSage models (Hamilton et al., 2017) on real and synthetic data, respectively, and then evaluate them on the test set consisting entirely of real data. This allows us to compare the performance for any relational database with a time component, without the need for manual data processing or feature engineering steps, which might introduce bias or noise. Notably, this is the only approach to relational utility that includes the whole relational database, ensuring a more comprehensive evaluation. We call this approach **relational deep learning utility (RDL-utility)**.

4 BENCHMARKING AND RESULTS

We combine our findings into a synthetic relational database benchmark, including single column, single-table, and multi-table fidelity metrics, alongside our novel approach to evaluating the utility of relational synthetic data, GNN-utility.

We compare the following methods for synthesizing relational data: **SDV**, **RC-TGAN**, **REaLTabFormer**, **ClavaDDPM**, **RGCLD**, and **TabularARGN**. Other related work does not have an API or available source code or we were not able to run the source code.

We include 6 datasets that feature in related work (**AirBnB**, **Rossmann**, **Walmart**, **Biodegradability**, **MovieLens**), the **Cora** dataset by McCallum et al. (2000), a popular dataset in graph representation learning, and the **F1** dataset from the relational deep learning benchmark Robinson et al. (2024) for a total of **8 benchmark datasets**. The datasets vary in types of relationships and number of tables and columns, which are summarized in Table 1 (see Appendix B.2 for details).

Table 1: **Summary of the 8 benchmark datasets.** The number of columns represents the number of non-id columns. The collection is diverse and covers all types of relational structures.

Dataset Name	# Tables	# Rows	# Columns	# Relationships	Hierarchy Type
Rossmann	2	59,085	16	1	Linear
AirBnB	2	57,217	20	1	Linear
Walmart	3	15,317	17	2	Multi Child
Cora	3	57,353	2	3	Multi Child
Biodegradability	5	21,895	6	5	Multi Child & Parent
IMDB MovieLens	7	1,249,411	14	6	Multi Child & Parent
Berka	8	757,722	37	8	Multi Child & Parent
F1	9	74,063	33	13	Multi Child & Parent

We evaluate all three levels of synthetic relational databases, with a focus on multi-table evaluation (see Appendix B.1 for all benchmark metrics). Most methods are non-deterministic, so we report results for three different replications. However, all results are stable across replications. Four of the methods are capable of synthesizing all of the datasets, irrespective of their relational structure. REALTABFORMER is only capable of generating databases with linear structure, and ClavaDDPM is unable to model datasets with two or more foreign keys between a pair of tables (*Biodegredability* and *CORA*).

For single-column metrics, we report the complement of the Kolmogorov-Smirnov statistic and the Total Variation Distance (the complement to KS/TV distance between two distributions P and Q is $1 - D_{\text{KS/TV}}(P||Q)$). For single-table metrics, we report the complements of the KS and TV distances between column pair correlations. For multi-table metrics, we report the average cardinality shape similarity and the k-hop ($k > 1$) correlations between tables. At all levels, we report the C2ST using XGBoost Chen & Guestrin (2016) as the discriminative model. We use 5-fold stratified cross-validation to estimate detection accuracy. For C2ST-Agg, we augment the rows with (a) counts of child rows for each row in each parent table, (b) the mean values of the numeric columns in the child table corresponding to the parent row; and (c) the number of unique categories in related rows.

4.1 Single Table Performance

We first evaluate the fidelity of individual tables. We focus on the detection metric and how well column pairs (bivariate distributions) are modeled. Table 2 summarizes the results. On the datasets that it is able to generate, the diffusion-based ClavaDDPM performs best, followed by the other diffusion-based method, RGCLD.

Table 2: **Single-table results**. For each dataset and metric we report the average detection accuracy (C2ST - lower is better) and column pair trends (Pairs - higher is better) across all tables for three independent samples. SDV exceeds the time limit for sampling on IMDB (TLE) and "-" denotes a method is unable to generate the dataset. The best result is **bolded** and the second-best <u>underlined</u>.

Dataset	Metric	TabularARGN	RGCLD	ClavaDDPM	RCTGAN	REALTABF.	SDV
Airbnb	C2ST (↓)	**0.64 ± 3e-3**	<u>0.70 ± 0.06</u>	0.78 ± 6e-4	0.88 ± 3e-3	0.84 ± 0.08	≈ 1
	Pairs (↑)	**0.93 ± 0.01**	<u>0.89 ± 0.01</u>	0.88 ± 2e-3	0.79 ± 5e-3	0.54 ± 0.02	0.49 ± 1e-3
Rossmann	C2ST (↓)	**0.56 ± 0.01**	0.69 ± 0.07	<u>0.67 ± 2e-3</u>	0.88 ± 0.01	0.75 ± 0.01	0.97 ± 4e-3
	Pairs (↑)	**0.91 ± 1e-3**	<u>0.90 ± 0.01</u>	0.85 ± 0.01	0.84 ± 0.01	0.85 ± 0.02	0.68 ± 4e-3
Walmart	C2ST (↓)	0.84 ± 0.01	<u>0.67 ± 0.03</u>	**0.54 ± 0.03**	0.76 ± 0.01	0.71 ± 0.02	0.87 ± 0.01
	Pairs (↑)	0.84 ± 4e-3	<u>0.92 ± 0.02</u>	**0.94 ± 2e-3**	0.87 ± 4e-3	0.83 ± 0.01	0.88 ± 4e-3
Berka	C2ST (↓)	0.72 ± 3e-3	<u>0.64 ± 0.04</u>	**0.54 ± 2e-3**	0.68 ± 0.01	-	0.82 ± 0.01
	Pairs (↑)	0.70 ± 4e-3	0.74 ± 0.03	**0.89 ± 0.02**	<u>0.74 ± 5e-3</u>		0.64 ± 2e-3
F1	C2ST (↓)	0.82 ± 0.01	**0.70 ± 0.02**	<u>0.71 ± 0.01</u>	0.81 ± 0.01	-	0.90 ± 4e-3
	Pairs (↑)	0.81 ± 0.01	**0.92 ± 0.01**	0.85 ± 9e-4	<u>0.90 ± 5e-4</u>		0.73 ± 3e-3
IMDB	C2ST (↓)	<u>0.51 ± 4e-3</u>	0.55 ± 0.04	**0.50 ± 1e-3**	0.55 ± 2e-3	-	TLE
	Pairs (↑)	<u>0.98 ± 2e-3</u>	0.94 ± 0.04	**0.99 ± 2e-3**	0.82 ± 6e-4		
Biodegradability	C2ST (↓)	<u>0.59 ± 3e-3</u>	0.63 ± 0.04	-	**0.58 ± 3e-3**	-	0.69 ± 2e-3
	Pairs (↑)	0.75 ± 0.01	<u>0.91 ± 0.09</u>		0.85 ± 0.04		**0.98 ± 0.01**
Cora	C2ST (↓)	<u>0.51 ± 0.01</u>	0.53 ± 0.02	-	**0.49 ± 2e-3**	-	0.75 ± 3e-3

The rankings of methods for single column metrics are similar to those of single tables. As expected, the methods model individual columns better. See Appendix C.1 for details. Interestingly, TabularARGN performs best on modeling marginal distributions, indicating their discretization approach is a robust preprocessing step.

4.2 Multi-Table Performance

Multi-table metrics examine how well the relationship cardinality and the relationships between different tables are preserved. Cardinality shape similarity examines only the former, while k-HOP similarity evaluates the latter; C2ST-Agg examines both. We report the results in Table 3. As with single table fidelity, the diffusion-based methods perform best.

An important advantage of detection with aggregation is that for the most part it uses same features as when evaluating individual tables (with the exception of the aggregation attributes). This allows us to directly compare single and multi-table performance to see how well the methods model relational data. We examine the difference between single-table and multi-table detection-based fidelity. For most methods, we observe a significant drop in fidelity when adding aggregations. Figure 1 shows how C2ST detection accuracy increases when incorporating aggregations (i.e. information about relationships between tables). We investigate this further using explainability methods and show how these can be used to "debug" generative methods in Appendix C.2.

4.3 Relational Deep Learning Utility Performance

Table 4 summarizes the RDL-utility results (Section 3.2). The evaluation is performed on four datasets which contain a temporal feature, which is necessary for defining train and test splits. Details of the tasks are provided in the Appendix D. The utility scores follow the same trend as the multi-table results, with models achieving high fidelity also performing best in utility tasks. Comparison with naive baselines demonstrates that the models can learn from synthetic databases.

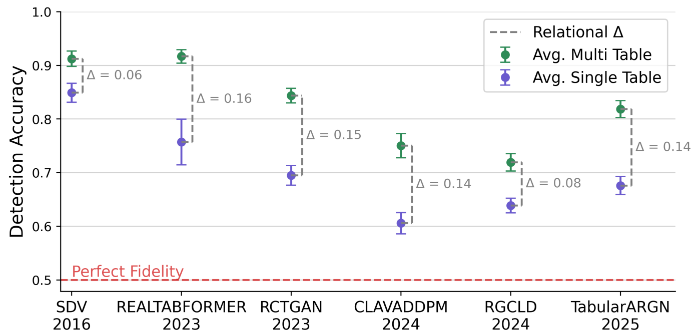


Figure 1: **Comparing single and multi-table performance**. While there is an overall trend in improvement of both single-table and multi-table fidelity, most methods still exhibit a significant gap between single-table and multi-table fidelity.

Table 3: **Multi-table results.** For each dataset and metric we report the average detection accuracy for C2ST-Agg (lower is better), cardinality similarity (higher is better) and k-hop correlation similarity (higher is better) across all tables for three independent samples. ”-” denotes a method is unable to generate the dataset, and TLE timeout. The best result is **bolded** and second-best underlined.

Dataset	Metric	TabularARGN	RGCLD	ClavaDDPM	RCTGAN	REALTABF.	SDV
Airbnb	C2ST-Agg (↓)	$\mathbf{0.63 \pm 0.02}$	$\underline{0.80 \pm 0.09}$	≈ 1	$0.98 \pm$ 1e-3	$0.99 \pm$ 4e-4	≈ 1
	Cardinality (↑)	0.99 ± 0.01	$\underline{0.99 \pm \text{2e-3}}$	$\mathbf{\approx 1}$	0.95 ± 0.01	0.76 ± 0.01	$0.26 \pm$ 5e-4
	1-HOP (↑)	0.80 ± 0.01	$\underline{0.84 \pm 0.05}$	$\mathbf{0.87 \pm \text{2e-3}}$	0.69 ± 0.01	0.34 ± 0.10	$0.25 \pm$ 5e-4
Rossmann	C2ST-Agg (↓)	$\mathbf{0.60 \pm 0.01}$	$\underline{0.76 \pm 0.01}$	$0.86 \pm$ 1e-3	0.86 ± 0.02	0.86 ± 0.02	$0.98 \pm$ 4e-3
	Cardinality (↑)	0.94 ± 0.03	0.99 ± 0.01	$\mathbf{0.99 \pm 0.01}$	0.83 ± 0.03	0.42 ± 0.18	$\underline{0.99 \pm \text{3e-3}}$
	1-HOP (↑)	$\mathbf{0.93 \pm 0.01}$	$\underline{0.88 \pm 0.01}$	0.83 ± 0.01	$0.87 \pm$ 3e-3	0.80 ± 0.01	0.74 ± 0.01
Walmart	C2ST-Agg (↓)	0.95 ± 0.03	0.89 ± 0.02	$\mathbf{0.73 \pm 0.05}$	0.95 ± 0.03	0.90 ± 0.02	$\underline{0.89 \pm 0.03}$
	Cardinality (↑)	0.66 ± 0.03	$\mathbf{0.95 \pm 0.01}$	$\underline{0.93 \pm 0.04}$	0.88 ± 0.03	0.86 ± 0.08	0.86 ± 0.02
	1-HOP (↑)	0.75 ± 0.03	$\underline{0.82 \pm 0.05}$	$\mathbf{0.86 \pm 0.03}$	$0.79 \pm$ 3e-3	$0.75 \pm$ 3e-3	0.77 ± 0.02
Berka	C2ST-Agg (↓)	0.81 ± 0.03	$\underline{0.73 \pm 0.04}$	$\mathbf{0.69 \pm 0.01}$	0.77 ± 0.04	-	$0.77 \pm$ 2e-3
	Cardinality (↑)	0.85 ± 0.01	≈ 1	$\underline{0.96 \pm 0.01}$	0.81 ± 0.02		0.81 ± 0.01
	1-HOP (↑)	0.73 ± 0.01	$\underline{0.81 \pm 0.03}$	$\mathbf{0.88 \pm 0.03}$	0.79 ± 0.02		0.59 ± 0.01
	2-HOP (↑)	0.66 ± 0.01	0.74 ± 0.03	$\mathbf{0.84 \pm 0.04}$	$\underline{0.78 \pm 0.02}$		$0.23 \pm$ 4e-3
	3-HOP (↑)	0.59 ± 0.01	0.65 ± 0.09	$\mathbf{0.81 \pm 0.04}$	$\underline{0.79 \pm 0.01}$		0.58 ± 0.01
F1	C2ST-Agg (↓)	0.96 ± 0.02	$\mathbf{0.74 \pm 0.05}$	$\underline{0.83 \pm \text{4e-3}}$	0.91 ± 0.01	-	$0.95 \pm$ 4e-3
	Cardinality (↑)	0.58 ± 0.06	$\mathbf{\approx 1}$	$\underline{0.88 \pm 0.05}$	0.57 ± 0.03		$0.72 \pm$ 2e-3
	1-HOP (↑)	$0.77 \pm$ 4e-3	$\mathbf{0.88 \pm 0.03}$	$\underline{0.79 \pm \text{5e-4}}$	0.79 ± 0.01		$0.68 \pm$ 3e-3
	2-HOP (↑)	0.76 ± 0.01	$\mathbf{0.88 \pm 0.03}$	$\underline{0.84 \pm \text{2e-3}}$	0.83 ± 0.01		$0.77 \pm$ 4e-3
IMDB	C2ST-Agg (↓)	0.74 ± 0.03	$\underline{0.65 \pm 0.10}$	$\mathbf{0.65 \pm 0.01}$	0.82 ± 0.03	-	TLE
	Cardinality (↑)	0.81 ± 0.01	$\mathbf{\approx 1}$	$\underline{0.99 \pm \text{6e-4}}$	0.80 ± 0.02		
	1-HOP (↑)	$\underline{0.89 \pm 0.01}$	0.86 ± 0.09	$\mathbf{0.92 \pm 0.02}$	$0.82 \pm$ 3e-3		
Biodegradability	C2ST-Agg (↓)	$0.89 \pm$ 4e-3	$\mathbf{0.71 \pm 0.07}$	-	$\underline{0.84 \pm 0.06}$	-	$0.98 \pm$ 1e-3
	Cardinality (↑)	$0.80 \pm$ 4e-3	$\mathbf{0.98 \pm \text{2e-4}}$		$\underline{0.85 \pm 0.01}$		0.61 ± 0.01
	1-HOP (↑)	0.61 ± 0.01	$\mathbf{0.78 \pm 0.08}$		$\underline{0.76 \pm 0.03}$		0.49 ± 0.01
	2-HOP (↑)	0.61 ± 0.01	$\underline{0.75 \pm 0.07}$		$\mathbf{0.77 \pm 0.03}$		0.48 ± 0.04
Cora	C2ST-Agg (↓)	$\underline{0.69 \pm 0.01}$	$\mathbf{0.62 \pm 0.03}$	-	0.74 ± 0.01	-	≈ 1
	Cardinality (↑)	$\underline{0.96 \pm \text{2e-3}}$	$\mathbf{\approx 1}$		0.90 ± 0.04		0.69 ± 0.01
	1-HOP (↑)	$\mathbf{0.80 \pm 0.01}$	0.64 ± 0.05		$\underline{0.68 \pm \text{1e-3}}$		$0.05 \pm$ 2e-3

Table 4: **RDL-utility results.** We include 5 datasets that have a temporal feature. We report ROC-AUC (higher is better) for classification and MAE (lower is better) for regression tasks. We report the naive baseline scores (mean or majority class) in parenthesis. ”-” denotes that the utility pipeline could not be used due to poorly generated time columns. The **best** and second results are highlighted.

Dataset		ORIGINAL	TabularARGN	RGCLD	CLAVADDPM	RCTGAN	REALTABF.	SDV
Rossmann	MAE	156 (324)	278 ± 4	$\mathbf{195 \pm 8}$	$\underline{196 \pm 2}$	217 ± 2	292 ± 21	$3,356 \pm 39$
Walmart	MAE	9531 (14.7k)	$13,844 \pm 40$	$\underline{13,165 \pm 264}$	$\mathbf{12,985 \pm 414}$	$13,681 \pm 194$	$14,275 \pm 180$	$13,830 \pm 131$
AirBnB	AUC	0.62 (0.5)	$\underline{0.61 \pm 0.02}$	0.60 ± 0.02	$\mathbf{0.61 \pm 0.02}$	0.54 ± 0.02	-	0.58 ± 0.00
Berka	AUC	0.73 (0.5)	$\underline{0.59 \pm 0.23}$	$\mathbf{0.60 \pm 0.01}$	0.52 ± 0.16	-	-	-
F1	AUC	0.76 (0.5)	0.40 ± 0.09	$\mathbf{0.74 \pm 0.03}$	$\underline{0.54 \pm 0.02}$	0.47 ± 0.05	-	0.48 ± 0.05

5 Conclusion

We surveyed methods for synthesizing relational databases and provided a critical review of approaches to evaluating the fidelity and utility of synthetic data. We integrated our findings into SyntheRela, the first benchmark tailored to evaluating synthetic *relational* databases.

We propose a robust detection-based metric for evaluating multi-table fidelity C2ST-Agg and a novel approach to evaluating the utility of synthetic relational databases, RDL-utility.

The best methods generate marginal distributions well, which is in line with SOTA single-table methods. Diffusion-based approaches, in particular, perform well on single-table fidelity. However, most methods experience a decline in performance when evaluated on multi-table fidelity, highlighting the added complexity of modeling relational databases. While no method perfectly preserves inter-table relationships, most outperform naive baselines and achieve model performance scores comparable to those trained on the original database. This suggests that despite not achieving perfect fidelity, i.e., being indistinguishable from real data, these synthetic data remain valuable for downstream machine learning tasks.

We hope our benchmark will provide valuable insights to synthetic data users and serve as a basis for comparison of novel synthetic relational database generation methods.

Acknowledgments

We would like to thank the Data Science initiative from the Faculty of Computer and Information Science at the University of Ljubljana and the Slovenian National Supercomputing Network (SLING) for granting us access to their hypercomputing cluster through the University of Ljubljana. We would like to clarify that we have no affiliation with any commercial entity.

References

Airbnb. Airbnb new user bookings. `https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings`, 2015.

Arno Appenzeller, Moritz Leitner, Patrick Philipp, Erik Krempel, and Jürgen Beyerer. Privacy and utility of private synthetic data for medical data analyses. *Applied Sciences*, 12(23):12320, 2022.

Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Ro(Beaulieu-Jones et al., 2019).bert E Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, pp. 1–8, 2020.

Brett K Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P Bhavnani, James Brian Byrd, and Casey S Greene. Privacy-preserving generative deep neural networks support clinical data sharing. *Circulation: Cardiovascular Quality and Outcomes*, 12(7):e005122, 2019.

Petr Berka et al. Guide to the financial data set. *PKDD2000 discovery challenge*, 2000.

Hendrik Blockeel, Sašo Džeroski, Boris Kompare, Stefan Kramer, Bernhard Pfahringer, and Wim Laer. Experiments in predicting biodegradability. *Applied Artificial Intelligence*, 18, 06 1999.

Daniel Bonnéry, Yi Feng, Angela K Henneberger, Tessa L Johnson, Mark Lachowicz, Bess A Rose, Terry Shaw, Laura M Stapleton, Michael E Woolley, and Yating Zheng. The promise and limitations of synthetic data as a strategy to expand access to state-level multi-agency longitudinal data. *Journal of Research on Educational Effectiveness*, 12(4):616–647, 2019.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Vadim Borisov, Kathrin Sessler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cEygmQNOeI.

Teodora Sandra Buda, Thomas Cerqueus, John Murphy, and Morten Kristiansen. Cods: A representative sampling method for relational databases. In *Database and Expert Systems Applications: 24th International Conference, DEXA 2013, Prague, Czech Republic, August 26-29, 2013. Proceedings, Part I 24*, pp. 342–356. Springer, 2013.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.

Kuntai Cai, Xiaokui Xiao, and Graham Cormode. Privlava: Synthesizing relational data with foreign keys under differential privacy. *Proc. ACM Manag. Data*, 1(2), jun 2023.

Luca Canale, Nicolas Grislain, Grégoire Lothe, and Johan Leduc. Generative modeling of complex data, 2022.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 785–794, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322.

Inc. DataCebo. *SDMetrics*, 10 2022. URL https://docs.sdv.dev/sdmetrics/. Version 0.8.0.

Asim Kumar Debnath, Rosa L. Lopez de Compadre, Gargi Debnath, Alan J. Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, 34(2):786–797, 1991.

Sašo Džeroski. *Relational data mining*. Springer, 2010.

F1. F1 db, 2021. URL https://github.com/f1db/f1db.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Relational deep learning: Graph representation learning on relational tables. *arXiv preprint arXiv:2312.04615*, 2023.

Will Cukierski FlorianKnauer. Rossmann store sales, 2015. URL https://kaggle.com/competitions/rossmann-store-sales.

Rainer Gemulla, Philipp Rösch, and Wolfgang Lehner. Linked bernoulli synopses: Sampling along foreign keys. In *Scientific and Statistical Database Management: 20th International Conference, SSDBM 2008, Hong Kong, China, July 9-11, 2008 Proceedings 20*, pp. 6–23. Springer, 2008.

Lise Getoor, Nir Friedman, Daphne Koller, Avi Pfeffer, and Ben Taskar. Probabilistic relational models. In *Introduction to Statistical Relational Learning*. The MIT Press, 08 2007.

Gretel.ai. Gretel blog. https://gretel.ai/blog, 2024. Accessed on March 24th, 2024.

Mohamed Gueye, Yazid Attabi, and Maxime Dumas. Row conditional-tgan for generating synthetic relational databases. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017. URL http://arxiv.org/abs/1706.02216.

Lasse Hansen, Nabeel Seedat, Mihaela van der Schaar, and Andrija Petrovic. Reimagining synthetic tabular data generation through data-centric AI: A comprehensive benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=dK1Rs1o0Ij.

F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4), dec 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL https://doi.org/10.1145/2827872.

Valter Hudovernik. Relational data generation with graph neural networks and latent diffusion models. In *NeurIPS 2024 Third Table Representation Learning Workshop*, 2024. URL https://openreview.net/forum?id=MNLR2NYN2Z.

Shingo Kato, suharay, Will Cukierski, and haisland0909. Coupon purchase prediction, 2015. URL https://kaggle.com/competitions/coupon-purchase-prediction.

Ilmun Kim, Aaditya Ramdas, Aarti Singh, and Larry Wasserman. Classification accuracy as a proxy for two-sample testing. *Annals of Statistics*, 49(1):411–434, 2021.

Stefan Kramer, Nada Lavrač, and Peter Flach. Propositionalization approaches to relational data mining. *Relational data mining*, pp. 262–291, 2001.

Jiayu Li and YC Tay. Irg: Generating synthetic relational databases using gans. *arXiv preprint arXiv:2312.15187*, 2023.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJkXfE5xx.

Ciro Antonio Mami, Andrea Coser, Eric Medvet, Alexander T. P. Boudewijn, Marco Volpe, Michael Whitworth, Borut Svara, Gabriele Sgroi, Daniele Panfilo, and Sebastiano Saccani. Generating realistic synthetic relational data through graph variational autoencoders, 2022.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

Anna Montoya, LizSellier, Meghan O'Connell, Wendy Kan, and alokgupta. Airbnb new user bookings, 2015. URL https://kaggle.com/competitions/airbnb-recruiting-new-user-bookings.

Mostly.ai. Mostlyai. https://mostly.ai/. Accessed on January 10th, 2025.

Jan Motl and Oliver Schulte. The ctu prague relational learning repository, 2024. URL https://arxiv.org/abs/1511.03086.

Eirini Ntoutsi, Pavlos Fafalios, Ujwal Gadiraju, Vasileios Iosifidis, Wolfgang Nejdl, Maria-Esther Vidal, Salvatore Ruggieri, Franco Turini, Symeon Papadopoulos, Emmanouil Krasanakis, et al. Bias in data-driven artificial intelligence systems—an introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(3):e1356, 2020.

Wei Pang, Masoumeh Shafieinejad, Lucy Liu, and Xi He. Clavaddpm: Multi-relational data synthesis with cluster-guided diffusion models. *arXiv preprint arXiv:2405.17724*, 2024.

Theodore Papamarkou, Tolga Birdal, Michael M. Bronstein, Gunnar E. Carlsson, Justin Curry, Yue Gao, Mustafa Hajij, Roland Kwitt, Pietro Lio, Paolo Di Lorenzo, Vasileios Maroulas, Nina Miolane, Farzana Nasrin, Karthikeyan Natesan Ramamurthy, Bastian Rieck, Simone Scardapane, Michael T Schaub, Petar Veličković, Bei Wang, Yusu Wang, Guowei Wei, and Ghada Zamzmi. Position: Topological deep learning is the new frontier for relational learning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39529–39555. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/papamarkou24a.html.

Neha Patki, Roy Wedge, and Kalyan Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, 2016.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Zhaozhi Qian, Bogdan-Constantin Cebere, and Mihaela van der Schaar. Synthcity: facilitating innovative use cases of synthetic data in different data modalities, 2023a. URL https://arxiv.org/abs/2301.07573.

Zhaozhi Qian, Rob Davis, and Mihaela van der Schaar. Synthcity: a benchmark framework for diverse use cases of tabular synthetic data. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023b. URL https://openreview.net/forum?id=uIppiU2JKP.

Pranav Rajpurkar, Emma Chen, Oishi Banerjee, and Eric J Topol. Ai in health and medicine. *Nature medicine*, 28(1):31–38, 2022.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan E. Lenssen, Yiwen Yuan, Zecheng Zhang, Xinwei He, and Jure Leskovec. Relbench: A benchmark for deep learning on relational databases, 2024. URL https://arxiv.org/abs/2407.20060.

Aivin V. Solatorio and Olivier Dupriez. Realtabformer: Generating realistic relational and tabular data using transformers, 2023.

Paul Tiwald, Ivona Krchova, Andrey Sidorenko, Mariana Vargas-Vieyra, Mario Scriminaci, and Michael Platzer. Tabularargn: A flexible and efficient auto-regressive framework for generating high-fidelity synthetic data, 2025. URL https://arxiv.org/abs/2501.12012.

Will Cukierski Walmart. Walmart recruiting - store sales forecasting, 2014. URL https://kaggle.com/competitions/walmart-recruiting-store-sales-forecasting.

gdantel Wendy Kan. Telstra network disruptions, 2015. URL https://kaggle.com/competitions/telstra-recruiting-network.

Sohier Dane World Bank. World development indicators, 2019. URL https://www.kaggle.com/datasets/theworldbank/world-development-indicators/data.

Kai Xu, Georgi Ganev, Emile Joubert, Rees Davison, Olivier Van Acker, and Luke Robinson. Synthetic data generation of many-to-many datasets via random graph generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Q120_4COf-K.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. *Advances in neural information processing systems*, 32, 2019.

EL Hacen Zein and Tanguy Urvoy. Tabular data generation: Can we fool XGBoost ? In *NeurIPS 2022 First Table Representation Workshop*, 2022. URL https://openreview.net/forum?id=tTQzJ6TJGVi.

Hengrui Zhang, Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4Ay23yeuz0.

APPENDIX

A A SURVEY OF SYNTHETIC RELATIONAL DATABASE GENERATION METHODS

The **Synthetic Data Vault (SDV)** (Patki et al., 2016) introduced the first learning-based method for generating relational databases. The method is based on the Hierarchical Modeling Algorithm (HMA) synthesizer, which is a multivariate version of the Gaussian Copula method. The method converts all columns to a predefined set of distributions and selects the best-fitting one. To learn dependencies, columns are converted to a standard normal before calculating the covariances. Tables are modeled with a recursive conditional parameter aggregation technique, which incorporates child table covariance and column distribution information into the parent table. The method requires the relational structure or metadata, which has since become a common practice.

The work of Mami et al. (2022) leverages the graph representation of relational database using **Graph Variational Autoencoders**. They focus on the case of one primary table connected by an identifier to an arbitrary number of secondary tables. The approach begins by transforming categorical, datetime, and numeric attributes into a normalised numeric format using an invertible function. Subsequently, all tables' attributes are merged into a single table, where rows from each table are vertically concatenated. This merged table, along with an adjacency matrix based on foreign key relations, forms a homogeneous graph representation of the dataset. Message passing is then applied to this graph representation using gated recurrent units (GRU). Following the message passing phase, the data is processed through a variational autoencoder, which encodes the joined table and random samples are taken from its latent space. These samples are then decoded back to the data space.

Composite Generative Models (Canale et al., 2022) propose a generative framework based on codecs for modeling complex data structures, such as relational databases. They define a codec as a quadruplet: C = (E,D,S,L), consisting of an encoder E producing embeddings and intermediate contexts, a decoder D for distribution representation, a sampler S and loss function L. The authors define the following codecs: Categorical and Numerical Codecs for individual columns, while composite data types are encoded using Struct and List Codecs, allowing for relational database synthesis. They also propose a specific implementation using causal transformers as generative models.

The **Row Conditional-TGAN (RC-TGAN)** (Gueye et al., 2023) extends the conditional tabular GAN model (Xu et al., 2019) to relational databases. RC-TGAN incorporates data from parent rows into the child table GAN model, allowing it to synthesise data conditionally on the connected parent table rows. The ability for conditional synthesis allows the method to handle various relationship schemas without additional processing. They enhance RC-TGAN to capture the influence of grandparent rows on their grandchild rows, preserving this connection even when the relationship information is not transferred by the parent table rows. Database synthesis is based on the row conditional generator of RC-TGAN model trained for each table. First, all parent tables are synthesised, followed by sampling the tables for which parents are already sampled. This allows using the synthesised parent rows as features when synthesizing child table rows.

The **Incremental Relational Generator (IRG)** (Li & Tay, 2023) uses GANs to incrementally fit and sample the relational dataset. They first define a topologically ordered sequence of tables in the dataset. Parent tables are modeled individually, while child tables undergo a three-step generation process. First, a potential context table is constructed by combining data from all related tables through join operations and aggregation. Then, the model predicts the number of child rows to be generated for each parent row, which they call its degree. They then extend the context table with corresponding degrees. Taking this table as context, they use a conditional synthetic tabular data generation model to generate the child table.

The **Realistic Relational and Tabular Transformer (REaLTabFormer)** (Solatorio & Dupriez, 2023) focuses on synthesizing single parent relational data and employs a GPT-2 encoder with a causal language model head to independently model the parent table. The encoder is frozen after training and used to conditionally model the child tables. Each child table requires a new conditional model, implemented as a sequence-to-sequence (Seq2Seq) transformer. The GPT-2 decoder with a causal language model head is trained to synthesise observations from the child table, accommodating arbitrary-length synthetic data conditioned on an input. While this method supports conditional synthesis of child rows, only one level is supported by this method.

Xu et al. (2023) propose a method for modeling many-to-many (M2M) datasets via random graph generation. They leverage a heterogeneous graph representation of the relational data and propose a factorization for modeling the graph representation incrementally. First, the edges of the graph are generated unconditionally using a random graph model. Second, one of the tables is generated conditionally on the topology of edges. One way to achieve such conditioning is by using a node embedding. Lastly, the remaining tables are generated using the conditional table model, which requires the generation of each node of the table based on the currently generated tables and all connections. They achieve this by using set embeddings to conditionally generate connected tables. The authors propose two variants using different conditional table models **BayesM2M** and **NeuralM2M**.

Privacy-preserving graphical models with latent variables. (PrivLava) (Cai et al., 2023) synthesizes relational databases with foreign key dependencies under differential privacy (DP). PrivLava models each foreign key in a relational schema as a separate graphical model, incorporating latent variables to capture inter-relational dependencies. Each entity in a child table associated with a parent table is modeled using a latent variable that represents the characteristics of the relationship. The approach handles foreign key relationships by treating them as a directed acyclic graph (DAG). It incrementally models the tables following a topological order based on the graph edges, beginning with root tables and then moving on to tables that depend on them. This ensures that each synthetic row in child tables is conditionally generated based on latent features of related parent rows. Gaussian oise is injected at various stages to achieve DP guarantees.

The **Cluster Latent Variable guided Diffusion Probabilistic Models (ClavaDDPM)** (Pang et al., 2024) utilizes classifier-guided diffusion models, integrating clustering labels as intermediaries between tables connected by foreign-key relations. The authors first propose a model for generating a single parent-child relationship. The connection between the tables is modeled by a latent variable obtained using Gaussian Mixture Model clustering. ClavaDDPM learns a diffusion process on the joint parent and latent variable distribution, followed by training a latent variable classifier on the child table to guide the diffusion model for the child table. Additionally, it includes a model to estimate child group sizes, to preserve relation cardinality. The authors then extend this to more parent-child constraints through bottom-up modeling and address multi-parent scenarios by employing majority voting to mitigate potential clustering inconsistencies.

Hudovernik (2024) adapts the tabular latent diffusion-based model TabSyn (Zhang et al., 2024) for conditional generation of relational databases. The method **Relational Graph-Conditioned Latent Diffusion (RGCLD)** utilizes a heterogenous graph representation of a relational database. The rows of each table are represented as nodes of a particular type, and the foreign keys between tables are represented by edges connecting the nodes. The method trains a graph neural network for each table to encode the relationships between connected tables. The embeddings of the GNN are used to guide the diffusion process in the latent space. During sampling, the method generates tables sequentially based on a topological order defined by the dataset's schema.

Tiwald et al. (2025) propose **TabularARGN**, an auto-regressive model for generating synthetic data for flat and sequential tables. TabularARGN is a shallow any-order auto-regressive network architecture. The method homogenizes diverse datatypes by discretizing them and is trained by minimizing the categorical cross-entropy loss across the discretized attributes. TabularARGN is designed to generate tabular synthetic data by learning the full set of conditional probabilities across features in a dataset. The sequential table TabularARGN model can handle sequences of arbitrary lengths. TabularARGN's sequential table model can utilize a flat table as context during training and generation, enabling the synthesis of two-table setups, such as a flat table containing time-independent information (e.g., bank customer data) and a sequential table containing time-dependent data (the bank customer transaction histories). The method was open-sourced by the commercial provider Mostly.ai, while the method is specialized for sequential data, they also support generating multi-parent schemas. For these datasets, the method will retain the context for one of the parent tables and retain the referential integrity for the rest[3].

[3]For details on multi-table generation see `https://mostly.ai/docs/generators/configure/set-table-relationships/multi-table`.

B Synthetic Relational Database Generation Benchmark

We provide our work as a Python package **SyntheRela**. The main goal of the package is the evaluation of the quality of synthetic relational databases. We can compare multiple methods across multiple databases with the *Benchmark* class or evaluate a single method on a single database with the *Report* class. All of the results of the benchmark are saved as JSON files and then parsed by our package for results summarization and visualization. The package is open source under the MIT license and can easily be extended with new methods, evaluation metrics, or databases.

B.1 Evaluation Metrics

We list the evaluation metrics for data fidelity and utility currently supported in our benchmark in Table 5, based on the granularity of the data they evaluate. For single column fidelity, we use the Kolmogorov-Smirnov and $\mathcal{X}^2$ statistical tests, total variation, Hellinger, Jensen-Shannon, and Wasserstein distances, alongside a single column C2ST. We include column pair correlation similarity, maximum mean discrepancy, pairwise correlation difference, and C2ST for single table fidelity. We evaluate multi-table fidelity with cardinality shape similarity and k-hop correlation similarity alongside C2ST-Agg. We also implement the Parent-Child C2ST for comparison with related work. Single-column utility is generally covered by fidelity metrics and not evaluated in related work. For single-table utility, we implement tabular machine learning utility metrics, and for multi-table utility, we include RDL-utility.

Table 5: **Evaluation metrics supported in the benchmark.**

	Single Column	**Single Table**	**Multi Table**
Statistical	KS Test, $\mathcal{X}^2$ Test	Column pair correlations	cardinality shape similarity
Distance	Total Variation, Hellinger, Jensen-Shannon, Wasserstein	Maximum Mean Discrepancy, Pairwise Correlation Difference	k-hop correlation similarity
Detection	C2ST	C2ST	C2ST-Agg, Parent-Child C2ST
Utility	/	Tabular ML-Utility	Relational DL-Utility

B.1.1 C2ST with Aggregation

Fidelity methods are concerned with measuring similarity between two databases with the same schema but different data. Typically, these will be the real database $\mathbb{D}_{\text{REAL}}$ and a synthetic database $\mathbb{D}_{\text{SYN}}$, with the goal of detecting if, to what extent, and where the synthetic data differ from the real data.

Let a relational database $\mathbb{D}$ be a collection of tables $\mathcal{T} = \{T_1, ..., T_n\}$ and a schema $\mathcal{S} = (\mathcal{R}, \mathcal{A})$, where $\mathcal{R} \subseteq \mathcal{T} \times \mathcal{T}$ are the relations between the tables and $A_{T_i} = \{a_1^{T_i}, \ldots, a_l^{T_i}\} \in \mathcal{A}$ define the tables' attributes. Each table is a set $T = \{v_1, ..., v_{n_T}\}$ consisting of elements v_i called rows. Each row $v \in T$ has three components $v = (p_v, \mathcal{K}_v, x_v)$. A **primary key** p_v that uniquely identifies the row v; the set of **foreign keys** $\mathcal{K}_v = \{p_{v'} : v' \in T'$ and $(T, T') \in \mathcal{R}\}$, where $p_{v'}$ is the primary key of the row v'; and the set of **values** $x_v = \{(a, x) : a \in A_T\}$ corresponding to attributes of table T.

Algorithm 1 describes how we add aggregations to the target table. For each child table, we add *CountRows*, a count of the number of child rows corresponding to a parent row. For each attribute (i.e. column) in each child table, we compute an aggregation attribute (*mean*, *count*, etc.). The aggregation attributes are added to the target table. In practice, different aggregation functions may be applied, as long as they maintain the i.i.d. assumption of the data. In our benchmark, we use column means for numerical columns and the number of distinct categories in categorical columns along with the child row count.

Our implementation allows us to directly control how many levels of aggregations we add (similarly to k-hop correlation similarity (Pang et al., 2024)). Each level accounts for one application of Algorithm 1. When aggregating for 1 level, only the information directly from the table's children is aggregated; at the second level, the aggregated grandchild columns are also added to the table.

Algorithm 1 Relational Aggregation.

Require: relational database $\mathbb{D}$ with tables $\mathcal{T}$ and relational schema $\mathcal{S} = \{\mathcal{R}, \{A_{T_1} \dots A_{T_n}\}\}$
Require: target table T
aggregationAttributes $\leftarrow [\,]$
for each $C_i \in \{C : (C, T) \in \mathcal{R}\}$ **do**
    Add *CountRows*(C_i, T) to aggregationAttributes ▷ Count number of child rows for each row in T.
    for each $a_j^{C_i} \in A_{C_i}$ **do** ▷ Iterate through child table columns.
        Add *Agg*$(C_i, a_j^{C_i}, T)$ to aggregationAttributes ▷ Compute aggregation (e.g., *mean*, *distinct*...).
    end for
end for
for each $v \in T$ **do**
    for each $\mathbf{a} \in$ aggregationAttributes **do**
        Add ($\mathbf{a}$.name, $\mathbf{a}$.value) to v ▷ Append computed aggregation attributes.
    end for
end for
return T_i ▷ Return table with aggregations applied.

In our benchmark, we choose to add only one level of aggregation, as most methods struggle to preserve the relationships at distance $k = 1$.

It has been shown that the accuracy-based approach to two-sample testing is consistent and controls for Type I error and (asymptotically) Type II error (see Kim et al. (2021) for theoretical results and a summary of empirical results). In practice, we are also interested in finite sample behavior. Experiment-based recommendations show that the approach should have an advantage in power when the data are well-structured or we have a lot of data, or when it is difficult to specify a test statistic, which is very common for high-dimensional data. Therefore, the large, higher-dimensional and structured nature of relational data is a perfect fit for C2ST.

B.2 Datasets

B.2.1 Relational Databases and Sampling Procedures

An important issue with evaluating relational data is that representative sampling is difficult (Buda et al., 2013; Gemulla et al., 2008). If the dataset does not include a time component or if the relationships are non-linear, the sampling becomes non-trivial and directly impacts the performance of the generative method. Even if the data have a strict hierarchy between tables, the rows in a child table are related via their parent, which breaks the assumption of iid sampling. This makes splitting the dataset into a representative train and test set difficult. Consequently, the methods for synthesizing relational databases are typically trained using the entire original dataset.

We organize the datasets used in related work based on the structure of their relational schema. Datasets using only linear relationships (one parent and one child table) include AirBnB (Montoya et al., 2015) and Rossmann Store Sales (FlorianKnauer, 2015). While this structure may be sufficient for some practical applications, Gueye et al. (2023) and Xu et al. (2023) highlight the need for methods supporting more complex, multiple-parent relational structures found in datasets like *MovieLens* (Harper & Konstan, 2015) and *World Development Indicators* (World Bank, 2019). Datasets including multiple child tables include *Telstra Network Disruptions* (Wendy Kan, 2015), *Walmart Recruiting - Store Sales Forecasting* (Walmart, 2014), and *Mutagenesis* (Debnath et al., 1991). Datasets with multiple children and parents include *Coupon Purchase Prediction* (Kato et al., 2015), *World Development Indicators* (World Bank, 2019), *MovieLens* (Harper & Konstan, 2015), *Biodegradability* (Blockeel et al., 1999) and *Berka* Berka et al. (2000).

B.2.2 Benchmark Datasets

Table 1 summarizes the relational datasets used in our benchmark. Six datasets are from related work and we add the *Cora* dataset by McCallum et al. (2000), which contains a simple yet challenging relational schema, and **F1** F1 (2021) from the RelBench benchmark. We include 2 datasets per hierarchy type to progressively add complexity in generation. The datasets used in our evaluation are diverse in terms of the number of columns, tables and relationships.

The **AirBnB** (Airbnb, 2015) dataset includes user demographics, web session records, and summary statistics. It provides data about users' interactions with the platform, with the aim of predicting the most likely country of the users' next trip. See Figure 6a for schema.

The **Berka** (also known as the Financial dataset) Berka et al. (2000) dataset contains 606 successful and 76 unsuccessful loans along with their information and transactions. The standard task is to predict the loan outcome for finished loans (A vs B) at the time of the loan start. See Figure 9 for schema.

The **Biodegradability** dataset (Blockeel et al., 1999) comprises a collection of chemical structures, specifically 328 compounds, each labeled with its half-life for aerobic aqueous biodegradation. This dataset is intended for regression analysis, aiming to predict the biodegradation half-live activity based on the chemical features of the compounds. See Figure 10 for schema.

The **Cora** dataset (McCallum et al., 2000) is a widely-used benchmark dataset in the field of graph representation learning. It consists of academic papers from various domains. The dataset consists of 2708 scientific publications classified into one of seven classes and their contents. The citation network consists of 5429 links. See Figure 7b for schema.

The **F1** dataset (F1, 2021) contains Formula 1 racing data and statistics dating back to 1950. It contains information on drivers, constructors, race results, and standings covering every season in F1 history. See Figure 8 for schema.

The **IMDB MovieLens** dataset (Harper & Konstan, 2015) comprises information on movies, actors, directors, and users' film ratings. The dataset consists of seven tables, each containing at least one additional feature besides the primary and foreign keys. See Figure 11 for schema.

The **Rossmann Store Sales** (FlorianKnauer, 2015) features historical sales data for 1115 Rossmann stores. The dataset consists of two tables connected by a single foreign key. This makes it the simplest type of relational dataset. The first table contains general information about the stores and the second contains sales-related data. See Figure 6b for schema.

The **Walmart** dataset (Walmart, 2014) includes historical sales data for 45 Walmart stores across various regions. It includes numerical, date-time and categorical features across three connected tables *store*,*features* and *depts*. The dataset is from a Kaggle competition, with the task of predicting department-wide sales. See Figure 7a for schema.

B.3 Comparison with Existing Evaluation Tools

The most popular and comprehensive package for evaluating tabular synthetic data is Synthcity (Qian et al., 2023a;b). It supports many statistical, privacy and detection-based (with several different models) metrics.

The only package that supports multi table evaluation is SDMetrics (DataCebo, 2022). It includes multi table metrics cardinality shape similarity and parent-child detection with logistic detection and support vector classifier. The package is not easy to extend and limits the adaptation of metrics. We re-implement detection metrics (discriminative detection, aggregation detection, and parent-child detection) to be used with an arbitrary classifier supporting the Scikit-learn (Pedregosa et al., 2011; Buitinck et al., 2013) classifier API. In SDMetrics, the results of different metrics are aggregated into a single-value, which limits the comparison of individual metrics between the methods and datasets. We re-implement the distance and statistical metrics so that each statistic, p-value, and confidence interval is easy to access.

Our benchmark package can be easily extended with new methods, metrics, and datasets. The process for adding custom metrics and new datasets is described in `https://github.com/martinjurkovic/syntherela/blob/main/docs/ADDING_A_METRIC.md`.

B.4 License and Privacy

We obtain the *AirBnB, Biodegradability, Cora, IMDB MovieLens, Rossmann and Walmart* datasets from the public SDV relational demo datasets repository (`https://docs.sdv.dev/sdv/single-table-data/data-preparation/loading-data`, accessed June 6th, 2024.). The SDV project is licensed under the Business Source License 1.1 (`https://github.com/`

sdv-dev/SDV?tab=License-1-ov-file#readme, which allows use for research purposes. The Berka dataset was obtained from the CTU Relational Dataset Repository (Motl & Schulte, 2024) an open-access repository of relational databases. The F1 dataset was obtained using the official RelBench implementation Robinson et al. (2024). It is released under the CC-BY-4.0 license (https://github.com/f1db/f1db?tab=CC-BY-4.0-1-ov-file). We remove all textual columns from the F1 dataset as relational generative methods are generally unable to generate those. We manually check all of the datasets to ensure they do not include any personally identifiable information. Some of the datasets contain processed columns, including aggregations of numerical values and connected table rows (e.g., nb_rows_in_{related table}). The authors of SDV (Patki et al., 2016) confirmed that these aggregations are not part of the original datasets, so we post-process all of the datasets to include only the columns found in their original form and update the metadata accordingly. We adapt some of the metrics from the SDMetrics (DataCebo, 2022) (MIT License) and Synthcity (Qian et al., 2023a;b) (Apache-2.0 License) synthetic data evaluation tools.

C Additional Experiments

C.1 Single Column Performance

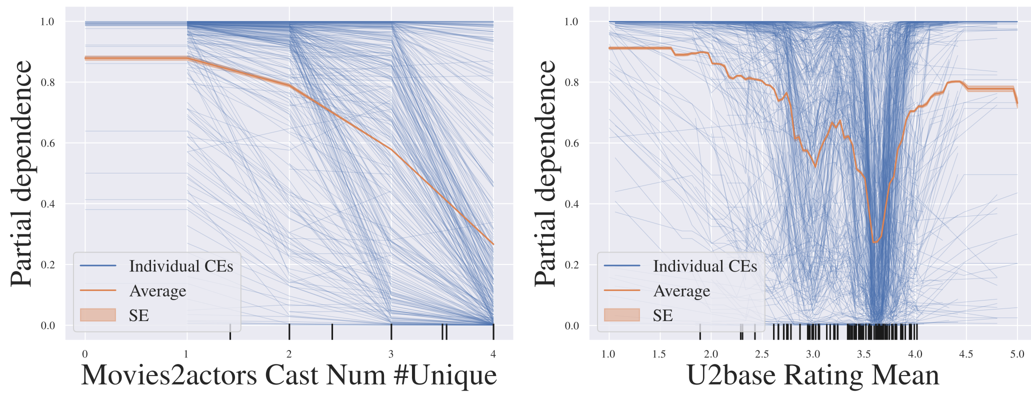
We evaluate the marginal distributions of individual columns by evaluating the column shapes and using the detection metric. Results are summarized in Table 6. TabularARGN performs best on modeling marginal distributions. We hypothesize this is a result of their preprocessing, which removes outliers and discretizes all columns. The method is closely followed by ClavaDDPM, which models individual tables best.

Table 6: **Single Column Results.** We report the detection accuracy for C2ST (lower is better), and the KS/TV complement for column shapes (higher is better). For each dataset and metric, we report the average across all tables for three independent samples. SDV exceeds the sampling time limit on IMDB (TLE) and "-" denotes a method is unable to generate the dataset. The best result is **bolded** and the second-best underlined.

Dataset	Metric	TabularARGN	RGCLD	ClavaDDPM	RCTGAN	REALTABF.	SDV
Airbnb	C2ST (↓)	$\mathbf{0.52 \pm 2e\text{-}3}$	$\underline{0.52 \pm 0.01}$	$0.55 \pm 4e\text{-}4$	$0.56 \pm 4e\text{-}4$	0.62 ± 0.01	$0.70 \pm 5e\text{-}4$
	Shapes (↑)	$\mathbf{0.96 \pm 9e\text{-}4}$	$\underline{0.96 \pm 0.01}$	$0.94 \pm 1e\text{-}4$	$0.89 \pm 3e\text{-}3$	0.72 ± 0.02	$0.59 \pm 8e\text{-}4$
Rossmann	C2ST (↓)	$\mathbf{0.51 \pm 3e\text{-}3}$	$\underline{0.53 \pm 0.01}$	$0.55 \pm 1e\text{-}3$	$0.55 \pm 4e\text{-}3$	$0.54 \pm 3e\text{-}3$	$0.60 \pm 2e\text{-}3$
	Shapes (↑)	$\mathbf{0.97 \pm 3e\text{-}3}$	$\underline{0.96 \pm 0.01}$	$0.94 \pm 1e\text{-}3$	$0.91 \pm 7e\text{-}4$	0.91 ± 0.01	$0.81 \pm 3e\text{-}3$
Walmart	C2ST (↓)	0.59 ± 0.01	0.59 ± 0.01	$\mathbf{0.53 \pm 0.01}$	0.64 ± 0.01	$\underline{0.59 \pm 0.01}$	0.67 ± 0.01
	Shapes (↑)	$\underline{0.89 \pm 0.01}$	0.88 ± 0.02	$\mathbf{0.92 \pm 0.01}$	0.82 ± 0.01	$\underline{0.82 \pm 0.01}$	$0.82 \pm 2e\text{-}3$
Berka	C2ST (↓)	$0.59 \pm 5e\text{-}3$	0.60 ± 0.02	$\mathbf{0.47 \pm 2e\text{-}3}$	$\underline{0.57 \pm 3e\text{-}3}$	-	$0.71 \pm 3e\text{-}3$
	Shapes (↑)	$\underline{0.82 \pm 4e\text{-}3}$	0.79 ± 0.04	$\mathbf{0.92 \pm 2e\text{-}3}$	$\underline{0.82 \pm 0.01}$		0.56 ± 0.01
F1	C2ST (↓)	$0.62 \pm 6e\text{-}4$	$\mathbf{0.58 \pm 0.01}$	$\underline{0.61 \pm 3e\text{-}3}$	$0.64 \pm 4e\text{-}3$	-	$0.80 \pm 4e\text{-}3$
	Shapes (↑)	0.85 ± 0.02	$\mathbf{0.91 \pm 0.03}$	$\underline{0.85 \pm 5e\text{-}3}$	$\underline{0.90 \pm 0.01}$		0.53 ± 0.01
IMDB	C2ST (↓)	$\underline{0.50 \pm 2e\text{-}3}$	0.53 ± 0.03	$\mathbf{0.50 \pm 1e\text{-}3}$	$0.54 \pm 1e\text{-}3$	-	TLE
	Shapes (↑)	$\underline{0.98 \pm 2e\text{-}3}$	0.94 ± 0.05	$\mathbf{0.99 \pm 8e\text{-}4}$	$0.93 \pm 2e\text{-}3$		
Biodegradability	C2ST (↓)	$\mathbf{0.55 \pm 0.01}$	0.62 ± 0.03	-	$\underline{0.58 \pm 0.01}$	-	0.64 ± 0.01
	Shapes (↑)	$\underline{0.91 \pm 2e\text{-}3}$	0.85 ± 0.06		$\mathbf{0.91 \pm 0.01}$		0.79 ± 0.01
Cora	C2ST (↓)	$\underline{0.51 \pm 0.01}$	0.53 ± 0.02	-	$\mathbf{0.49 \pm 2e\text{-}3}$	-	$0.75 \pm 3e\text{-}3$
	Shapes (↑)	$\underline{0.93 \pm 2e\text{-}3}$	0.88 ± 0.03		$\mathbf{0.96 \pm 3e\text{-}3}$		$0.50 \pm 3e\text{-}3$

C.2 Interpretability for Generative Method Diagnostics

One benefit of using a machine learning model to evaluate fidelity is that we can use it to "debug" our generative method. By examining the features the discriminative model utilizes to distinguish the synthetic data from the original and exploring the patterns it learns, we can discover which aspects of the data the generative method fails to model well. ML interpretability with feature importance reveals that methods struggle with preserving the relationships between columns across tables. Figure 2 shows an example of how aggregation attributes summarizing information about child table rows are the most discriminative features. We further examine two such attributes in Figure 3. The partial dependence plots of the first and fourth most important features from Figure 2 show how subsets of both categorical (Fig. 3a) and numerical (Fig. 3b) features' conditional distributions are informative to the discriminative model. We include the interpretability method directly into our implementation of the detection metric, allowing users to immediately investigate where their method is underperforming after evaluating the synthetic data.



(a) (b)

Figure 3: **Partial dependence plots**. Results are for the 1st and 4th most important feature from Figure 2. With ideally generated synthetic data, features could not discriminate between synthetic and original data and every partial dependence plot would be a horizontal line at 50% probability. We can observe that (a) the synthetic data have too many unique actor cast numbers (higher probability of being synthetic when feature value is larger than 4) and (b) the mean movie ratings in the original data vary more than in the synthetic data, where they are more concentrated around 3.5.

C.3 Shortcomings of Logistic Detection

As explained in Section 2.2 a significant limitation of LD is its inability to capture interactions between columns. It can thus assign a perfect fidelity score to a dataset that is completely corrupted. In this section, we empirically show this shortcoming. We conduct the experiment by selecting a table from each dataset (with the exception of CORA in which no table has two columns, which are not primary or foreign keys). We first select the table and split it in half to simulate the original table and a perfectly generated (by the underlying DGP) synthetic table. We then copy the "generated" table and randomly shuffle values in each column, completely ruining the fidelity of the dataset while keeping the marginal distributions intact. We then evaluate the perfectly generated and shuffled datasets using LD and C2ST using XGBoost. The results are visualized in Figure 4.

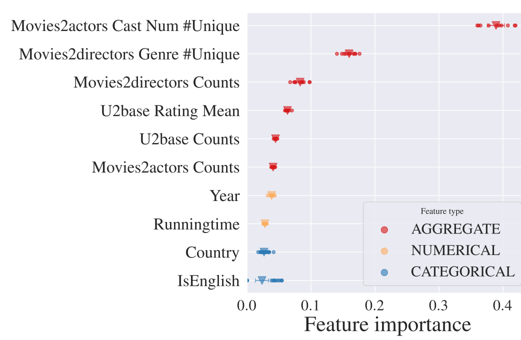


Figure 2: **Feature importance for C2ST with aggregation using XGBoost.** Results are for the best-performing method. The added features that incorporate relational information (red) are the most important for discriminating between real and synthetic data.

Notably LD assigns both versions of the dataset the same score, labeling them indistinctive from the original data. If the fidelity aspect of interest would be solely the marginal distributions, the LD results would be more appropriate than those of C2ST using XGBoost (as marginals are identical in both datasets). However, given that we are interested in single table fidelity, our experiment showcases a fundamental shortcoming of LD as a measure of single table fidelity.

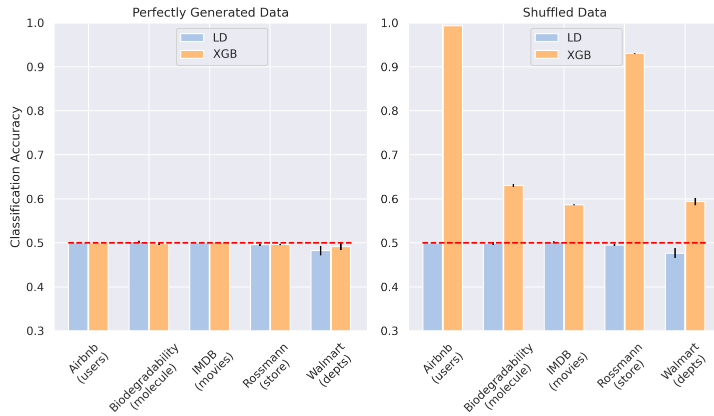


Figure 4: **Issues with logistic detection.** For each dataset, we simulate a perfectly generated table by splitting the original table in half. We copy one part of the table and shuffle the values in each column and thus completely ruin the fidelity of the table. While the XGBoost classifier can almost perfectly segment the corrupted rows, logistic regression assigns both of the datasets the same score.

C.4 Classifier Accuracy as a Data Copying Diagnostic

We investigate how the accuracy of the discriminative model in classifier two sample testing can be used to diagnose data copying in Figure 5. We also demonstrate how the classifier performance commonly reported in LD ($2 \cdot \max(\text{AUC}, \frac{1}{2}) - 1$) masks this issue.

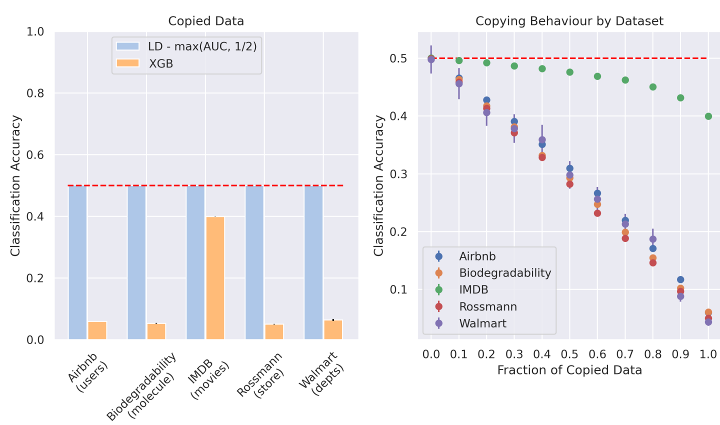


Figure 5: **Detecting data copying with C2ST.** The left plot demonstrates how the error estimation of LD ($2 \cdot \max(\text{AUC}, \frac{1}{2}) - 1$) masks data copying, while C2ST with XGBoost detects it across all datasets. In the right plot, we observe how copying only a fraction of the original data affects discriminative model accuracy, with accuracy consistently decreasing as more data is duplicated.

As in the previous experiment, we simulate a perfect synthetic generating a dataset by splitting the original table in half. However, instead of introducing corruption into the second half, we create an exact copy of the original data (i.e., the first half). The commonly used LD implementation fails to detect data copying and assigns the copied data a perfect score. In contrast, C2ST (with XGBoost) accuracy successfully detects data copying as accuracy drops significantly below 50%. We then examine the behaviour of C2ST when only a portion of the data is copied. We keep a portion of the dataset as an identical copy and sample the rest of the values from the ”perfectly generated” half. For most of the datasets, even when a relatively low percentage of the data is copied, the model detects the duplication.

D RELATIONAL DEEP LEARNING UTILITY

We incorporate the RelBench package into our benchmark to evaluate the utility of synthetic databases. We incorporate 5 of our databases that have a temporal feature. We explain the tasks for each of them below.

- **Rossmann**: We predict the number of sales for each store for each day.
- **Walmart**: We predict the weekly sales for each department of each store.
- **AirBnB**: We predict whether a user has already made a booking or not.
- **Berka**: We predict the binary status of the loan (successful or unsuccessful).
- **F1**: We predict whether a driver will qualify in the top-3 for a race in the next month. This is one of the original RelBench predictive tasks[4].

We use the GraphSage GNN model and train the model with the default RelBench hyperparameters.

E LIMITATIONS AND FUTURE WORK

Our work focused on fidelity and utility, but not privacy. While we do briefly touch upon one aspect of privacy - data copying - we delegate the research of privacy metrics for synthetic relational databases to future work. Finally, several aspects of synthetic data evaluation are limited by the difficulty of representative sampling. More work needs to be done in understanding the limitations and preparing new benchmark datasets or dataset generators.

F EXPERIMENTAL SETUP

F.1 COMPUTATIONAL RESOURCES

The generative methods were trained on NVIDIA 32GB V100S GPUs and H100 80GB GPUs. The total number of GPU hours spent across all experiments is approximately 500. Results which do not require a GPU were run on machines running AMD EPYC 7702P 64-Core Processor with 256GB of RAM. All experiments were performed on an internal HPC cluster.

F.2 REPRODUCIBILITY

F.2.1 DATASETS AND DATA SPLITTING

Scripts for downloading the datasets and their metadata in the SDV format (Patki et al., 2016) are available in the project repository, as well as the corresponding synthetic data samples for all methods to enable the reproduction of the benchmark results.

We opt not to split the datasets into train, test, and validation sets for generative model training. When no temporal information is included and the structure is non-linear the representative sampling in relational datasets is non-trivial. We delegate this to future work.

Due to computational limits (also reported by Solatorio & Dupriez (2023)), we subsample the Rossmann Store Sales, AirBnB, and Walmart datasets. Additionally, we subsample the Berka and F1 datasets for the purposes of obtaining a held-out test set for our utility experiments.

- **Rossmann Store Sales**: Subsampled on table *historical*, column *Date* by taking the rows of a two month period from *2014-07-31* to *2014-09-30*, similarly to Solatorio & Dupriez (2023).
- **AirBnB**: Subsampled the dataset by only including the users who have less than 50 sessions and then sampled 10k users, as done by Solatorio & Dupriez (2023).
- **Walmart**: Subsampled on tables *departments* and *features* on the column *Date* by taking the rows from January 2012.

[4] `https://relbench.stanford.edu/datasets/rel-f1#driver-top3`

- **Berka**: We use the data prior to *1998-01-01* and use the *1998* data as the test set.
- **F1**: We use the data prior to *2010-01-01* for training. This corresponds to the test set timestamp in the official RelBench implementation [5].

F.2.2 Experimental Details and Hyperparameters

To provide some quantification of the variability from the non-deterministic nature of the methods, we generated synthetic data for each of the methods for each of the datasets 3 times with different fixed random seeds. We ran the benchmark for each replication.

Scripts for reproducing the generative model training and instructions for training commercial methods are included in the project repository.

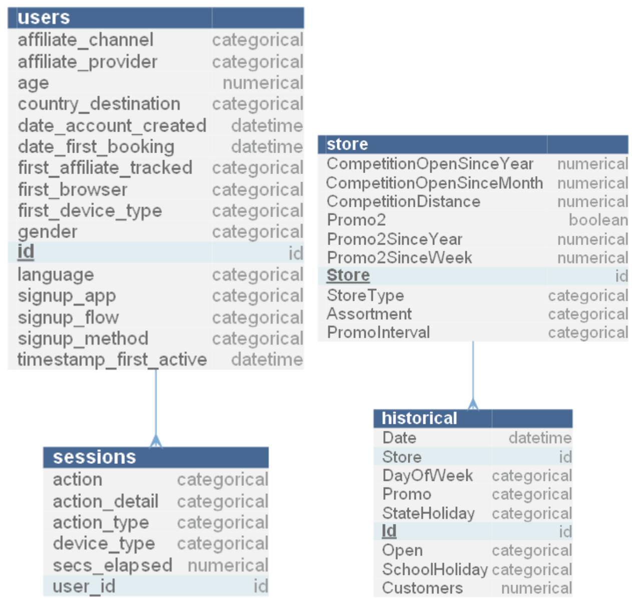
It is possible that better performance could be achieved by investing more effort into parameter tuning. However, due to our choice to not split the data, it was not clear how to optimize hyperparameters; therefore, we selected default hyperparameters for all methods (see Table 7).

[5] `https://relbench.stanford.edu/datasets/rel-f1/`

Table 7: **Hyperparameter specification.**

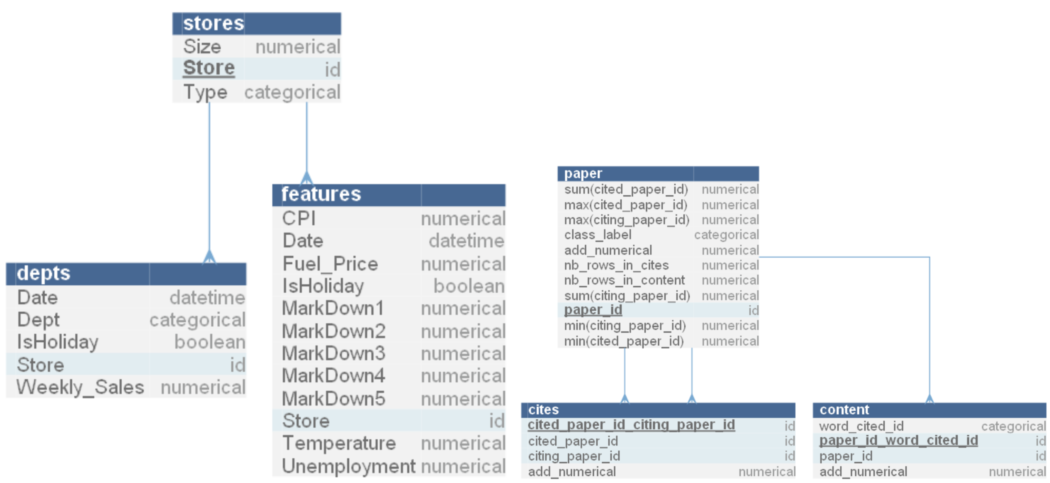
model	hyperparameter	value
RCTGAN	embedding_dim	128
	generator_dim	(256, 256)
	discriminator_dim	(256, 256)
	generator_lr	0.0002
	generator_decay	1e-06
	discriminator_lr	0.0002
	discriminator_decay	1e-06
	batch_size	500
	discriminator_steps	1
	epochs	1000
	pac	10
	grand_parent	True
	field_transformers	None
	constraints	None
	rounding	”auto”
	min_value	”auto”
	max_value	”auto”
SDV	locales	None
	verbose	True
	table_synthesizer	”GaussianCopulaSynthesizer”
	enforce_min_max_values	True
	enforce_rounding	True
	numerical_distributions	{}
	default_distribution	”beta”
REALTABFORMER	epochs	100
	batch_size	8
	train_size	0.95
	output_max_length	512
	early_stopping_patience	5
	early_stopping_threshold	0
	mask_rate	0
	numeric_nparts	1
	numeric_precision	4
	numeric_max_len	10
	evaluation_strategy	”steps”
	metric_for_best_model	”loss”
	gradient_accumulation_steps	4
	remove_unused_columns	True
	logging_steps	100
	save_steps	100
	eval_steps	100
	load_best_model_at_end	True
	save_total_limit	6
	optim	”adamw_torch”
TabularARGN (API)	Configuration presets	Accuracy
	Max sample size	100%
	Model size	Large
	Batch size	Auto
	Flexible generation	Off
	Value protection	Off
CLAVADDPM	num_clusters	50
	parent_scale	1.0
	classifier_scale	1.0
	num_timesteps	2000
	batch_size	4096
	layers_diffusion	[512, 1024, 1024, 1024, 1024, 512]
	iterations_diffusion	200000
	lr_diffusion	0.0006
	weight_decay_diffusion	1e-05
	scheduler_diffusion	”cosine”
	layers_classifier	[128, 256, 512, 1024, 512, 256, 128]
	iterations_classifier	20000
	lr_classifier	0.0001
	dim_t	128
RGCLD	GNN hidden dim	128
	GNN aggregation	sum
	GNN layers	# Tables
	GNN lr	0.008
	GNN weight decay	$1e-5$
	GNN epochs	250
	VAE layers	2
	VAE token dim	4
	VAE hidden dim	128
	VAE δ	0.7
	VAE (β_{max}, β_{min})	(0.01, $1e-5$)
	VAE lr	$1e-3$
	VAE epochs	4000
	Diff model dim	1024
	Diff lr	0.001
	Diff weight decay	$1e-6$
	Diff epochs	4000

G Dataset Schema



(a) AirBnB (b) Rossmann

Figure 6: Two table database diagrams.



(a) Walmart (b) Cora

Figure 7: Table database diagrams.

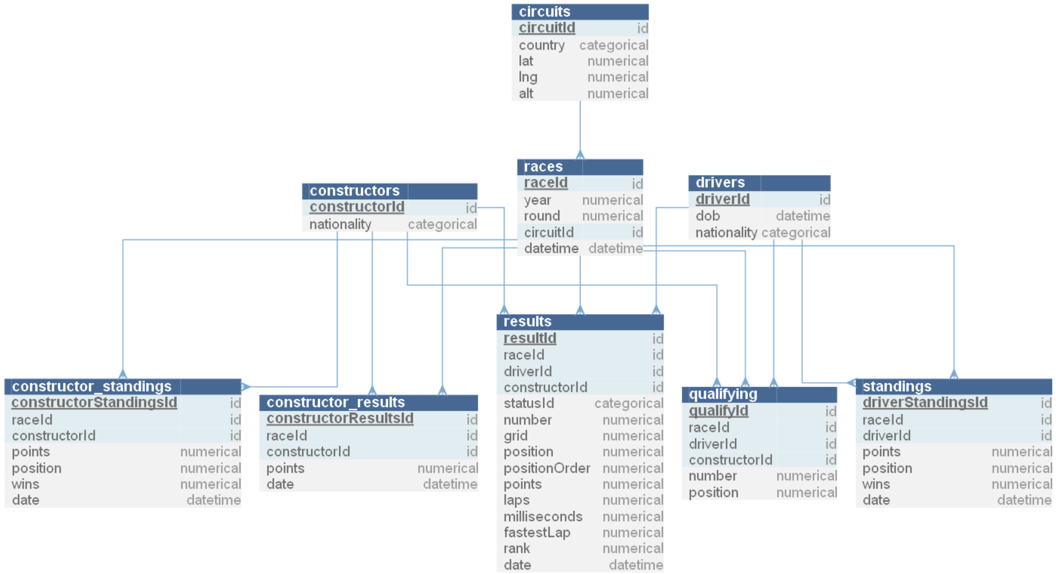


Figure 8: F1 database diagram.

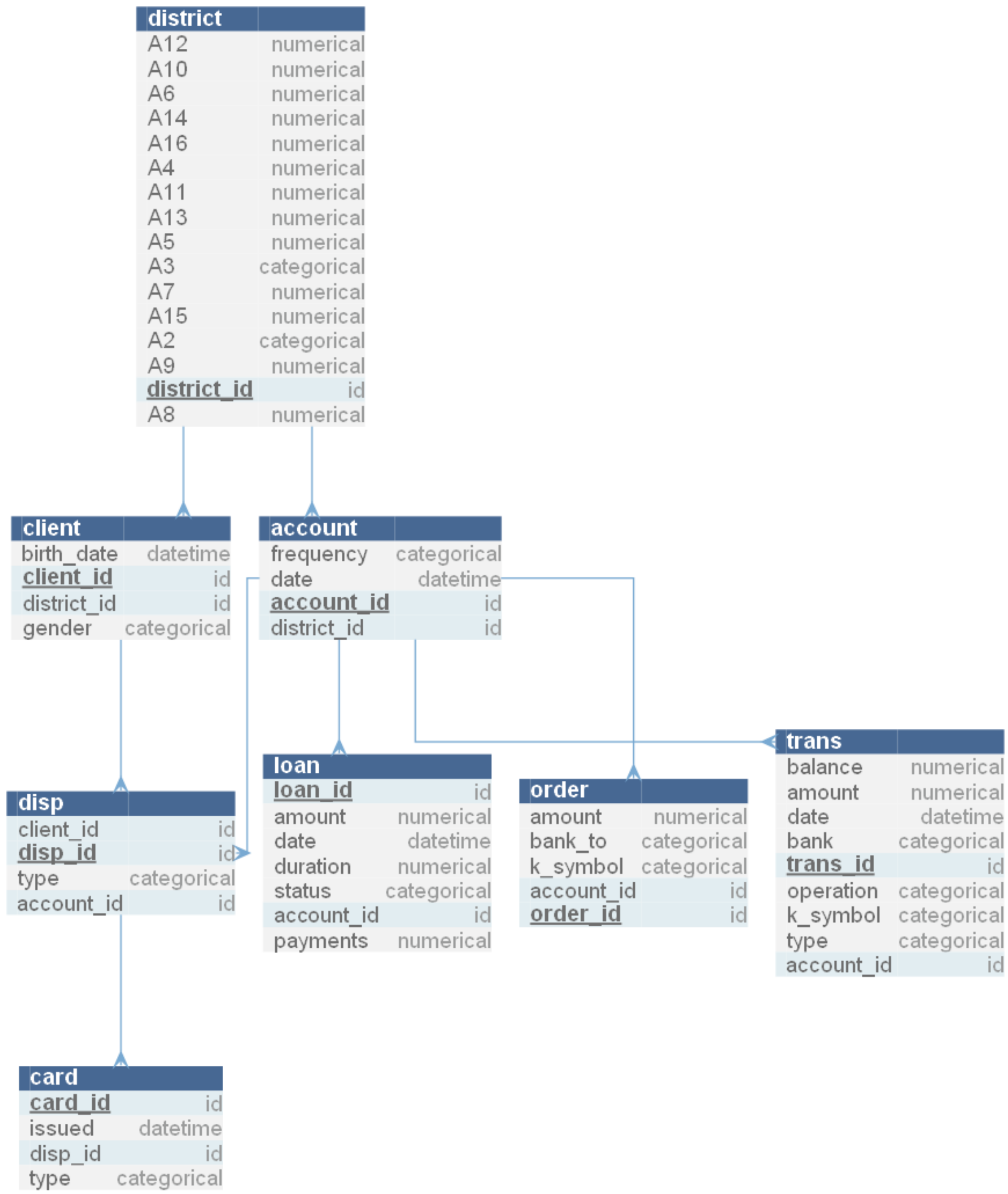


Figure 9: Berka database diagram.

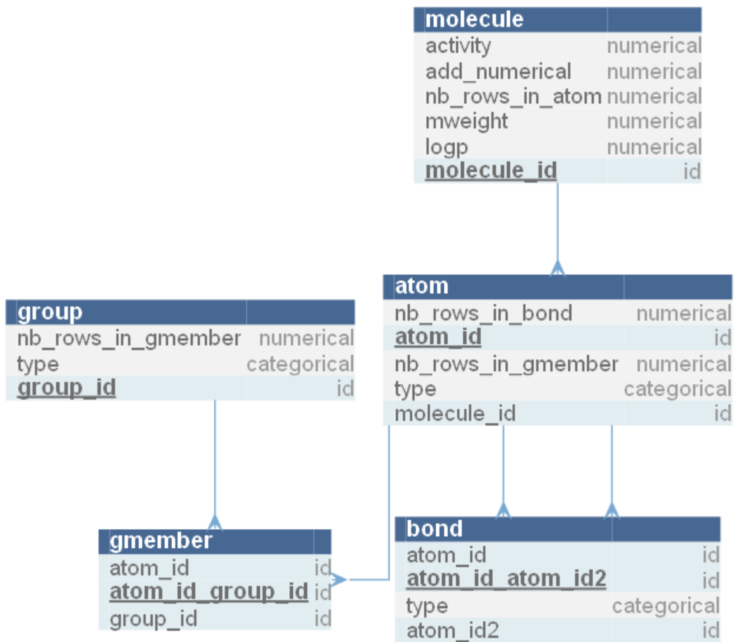


Figure 10: Biodegradability database diagram.

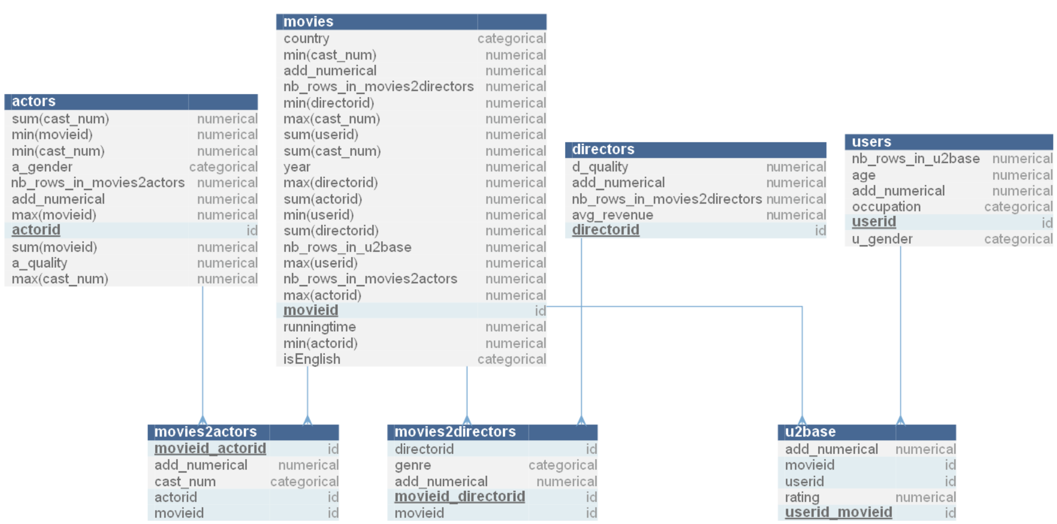


Figure 11: IMDB MovieLens database diagram.