Qymera

Simulating Quantum Circuits using RDBMS

Littau, Tim; Hai, Rihan

DOI
10.1145/3722212.3725126

Publication date
2025

Document Version
Final published version

Published in
SIGMOD-Companion 2025 - Companion of the 2025 International Conference on Management of Data

Qymera: Simulating Quantum Circuits using RDBMS

Tim Littau
Delft University of Technology
Delft, The Netherlands
t.m.littau@tudelft.nl

Rihan Hai
Delft University of Technology
Delft, The Netherlands
r.hai@tudelft.nl

Abstract

Quantum circuit simulation is crucial for quantum computing such as validating quantum algorithms. We present Qymera, a system that repurposes relational database management systems (RDBMSs) for simulation by translating circuits into SQL queries, allowing quantum operations to run natively within an RDBMS. Qymera supports a wide range of quantum circuits, offering a graphical circuit builder and code-based interfaces to input circuits. With a benchmarking framework, Qymera facilitates comparison of RDBMS-based simulation against state-of-the-art simulation methods. Our demonstration showcases Qymera's end-to-end SQL-based execution, seamless integration with classical workflows, and its utility for development, benchmarking, and education in quantum computing and data management. A video demonstrating Qymera's key features can be found at https://youtu.be/j_fTKnVIV6c.

CCS Concepts

• **Computing methodologies** → **Simulation tools**; **Quantum mechanic simulation**; • **Information systems** → *Relational database model*.

Keywords

Quantum Data Management, Quantum Computing, Simulation

ACM Reference Format:
Tim Littau and Rihan Hai. 2025. Qymera: Simulating Quantum Circuits using RDBMS. In *Companion of the 2025 International Conference on Management of Data (SIGMOD-Companion '25), June 22–27, 2025, Berlin, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3722212.3725126

1 Introduction

Quantum computing, heralded as the next frontier of computational science, promises revolutionary processing power beyond classical hardware. However, quantum computing remains in its early stages, known as the Noisy Intermediate-Scale Quantum (NISQ) era, where quantum hardware is limited by small numbers of qubits and noise. For instance, Heron[1], one of IBM's most performant quantum processors, has only 156 qubits and lacks full error correction. In this landscape, *simulation* serves as a core mechanism for exploring and validating quantum technologies, such as advancing quantum algorithm design, hardware development, and error correction [7].

[1]https://docs.quantum.ibm.com/guides/processor-types

SIGMOD-Companion '25, Berlin, Germany

ACM ISBN 979-8-4007-1564-8/2025/06
https://doi.org/10.1145/3722212.3725126

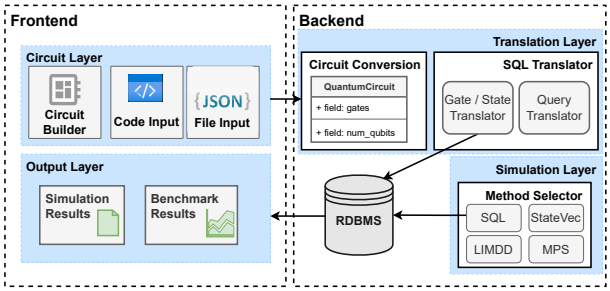


Figure 1: Overview of Qymera

Simulation refers to the use of classical computers to numerically reproduce the behavior of quantum circuits, tracking the evolution of quantum states and operations.

Existing simulation frameworks such as Qiskit[2], PyQuil[3], Cirq[4], typically access data relying on programming languages, e.g., Python. They require users to specify the simulation methods. To improve performance, low-level optimizations are needed, such as parallelization and Single-Instruction, Multiple-Data (SIMD) [8]. This approach can be cumbersome, especially for large-scale simulations, and places a high burden on developers to fine-tune performance.

Our proposal. We introduce a novel method that *utilizes relational database management systems (RDBMSs) for simulating quantum circuits*. In this approach, quantum circuits—comprising a series of quantum gates applied to qubits—are represented by storing qubit and gate information in tables. The simulation process is then driven by SQL queries that specify the required computations. By leveraging the declarative nature of RDBMSs, simulation engineers and database researchers can concentrate on defining the desired outcomes without delving into the underlying execution details. RDBMSs then handle the complexity by automatically optimizing and executing these queries through techniques such as logical and physical query planning, caching, and hardware-aware techniques such as parallelism [5].

However, using RDBMSs as an out-of-the-box solution for simulation poses quantum-specific challenges. First, quantum circuits often involve bit-level operations (e.g., manipulating specific qubits), which SQL does not natively support. Second, although RDBMS-based simulation can outperform existing simulation methods for certain circuits, they are not universally optimal [3]. In our preliminary experiments with a 2.0 GB memory limit, RDBMS approach simulated up to 3,118× more qubits than a conventional simulation method for sparse circuits but performed 14% worse on dense circuits (see appendix B4, Fig. 10 in [4]). Consequently, it is critical to identify scenarios where RDBMSs excel by benchmarking them

[2]Aer: https://qiskit.github.io/qiskit-aer/
[3]https://pyquil-docs.rigetti.com/en/stable/
[4]qsim: https://quantumai.google/qsim/cirq_interface

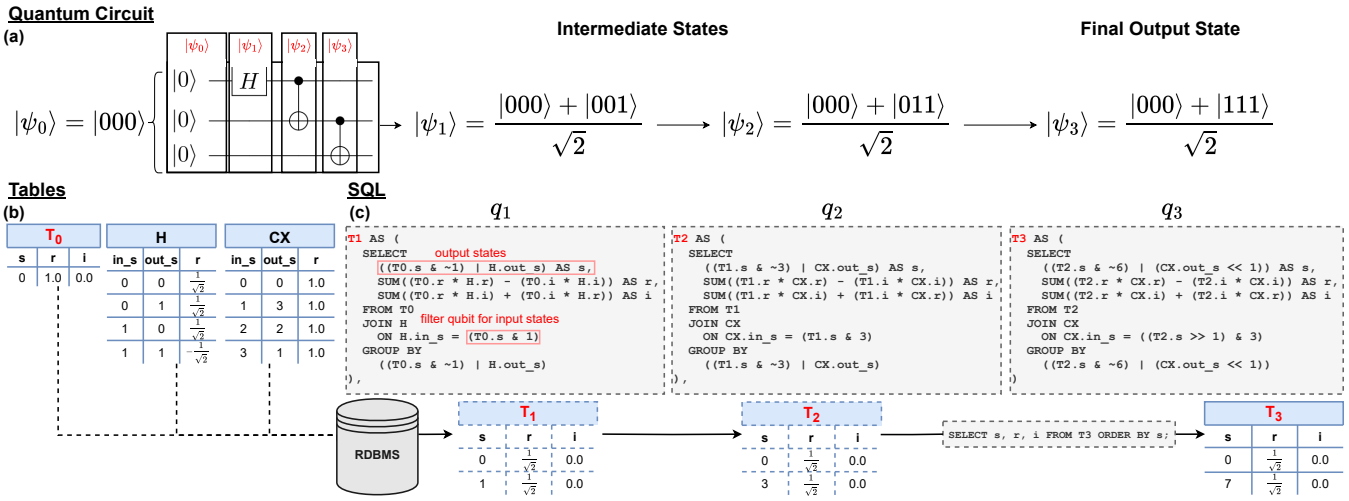


Figure 2: Running example of the SQL translation flow: (a) the input circuit transforming initial state $|\psi\rangle_0$ into final state $|\psi\rangle_3$, where horizontal wires represent qubits, and boxes indicate initial, intermediate, and final states; (b) tables for the initial state $|\psi\rangle_0$, the H gate, and the CNOT gate; and (c) SQL queries $\{q_1, q_2, q_3\}$ generated for each gate operation in the input circuit, along with the resulting intermediate tables T_1, T_2 and final output table T_3.

against existing simulation methods, rather than applying them blindly.

We present *Qymera*, a novel system for simulating quantum circuits using RDBMSs. *Qymera* translates quantum circuits into SQL queries, enabling RDBMS-based simulations. Its key features include: 1) a graphical circuit builder for intuitive design and parameterized circuits via Qiskit- or PyQuil-like syntax, 2) a unique feature of supporting RDBMS-based simulation, alongside state-of-the-art simulation methods such as state-vector, tensor networks, e.g., MPS, and LIMDD.[5] and 3) a benchmarking suite for systematically comparing RDBMS performance against alternative simulators on a wide range of circuit inputs.

2 Transforming Quantum Circuits into SQL

A quantum circuit consists of a sequence of gates, which operate on quantum states. Simulating quantum circuits involves translating the abstract mathematical representation of quantum states and gates into computational operations.

In this section, we explain how Qymera maps quantum states and gates to tables and translates circuit operations into SQL queries while addressing quantum-specific challenges. We walk through the SQL translation process using a running example in Fig. 2.

2.1 From Quantum States and Gates to Tables

An n-qubit quantum state is represented as

$$|\psi\rangle = \sum_{j=0}^{2^n-1} (a_j + i\, b_j)\, |j\rangle\,,$$

where j is an integer index ranging from 0 to $2^n - 1$, $|j\rangle$ is the computational basis state (binary strings of length n, such as $|0\cdots0\rangle$ to $|1\cdots1\rangle$). Each basis state $|j\rangle$ has a complex amplitude $a_j + ib_j$, where a_j and b_j are the real and imaginary parts, respectively.

We define the relational schema for a state $|\psi\rangle$ as `T(s, r, i)`, where s is the computational basis state $|j\rangle$ encoded as an integer, and r and i represent the real and imaginary parts of its complex amplitude, both being real-valued. Only nonzero basis states are stored.

A quantum gate transforms an input quantum state into an output state. We represent this transformation in a relational table that captures the integer indices of the input and output states, along with their transition amplitudes. We define the relational schema for a gate as `T(in_s, out_s, r, i)`, where `in_s` and `out_s` denote input and output state indices, and `r` and `i` represent the real and imaginary parts of the transition amplitude.

Example 2.1 (Quantum states and gates as tables). *Consider the circuit shown in Fig. 2a), generating a 3-qubit GHZ state. Initially, the state is* $|000\rangle$*, represented as* `(0, 1.0, 0)`*. A Hadamard gate mapping input indices to outputs, each with amplitude* $\frac{1}{\sqrt{2}}$ *applied to the first qubit transforms it into the superposition* $\frac{1}{\sqrt{2}}(|000\rangle + |100\rangle)$*, stored in the relational form as tuples* `(0,` $\frac{1}{\sqrt{2}}$`, 0)` *and* `(1,` $\frac{1}{\sqrt{2}}$`, 0)`*. Subsequent CX gates manipulate the state further by updating the table to represent resulting intermediate states* $|\psi\rangle_2$ *and* $|\psi\rangle_3$*.*

2.2 From Simulation Operations to SQL

Simulating quantum circuits requires bit-level manipulation, since each gate operates on specific qubits. To enable this in SQL, we adopt the previously explained *integer encoding* and apply *bitwise operations* (see Table 1) to pinpoint the relevant qubit and perform bit-level gate operations.

Example 2.2 (H Gate Operation via SQL). *Query* q_1 *in Fig. 2c) represents the Hadamard gate operation from Fig. 2a). Since the gate acts on the first qubit, we locate it in* `T0` *using* `T0.s & ~1` *and join*

[5]These simulation methods (also known as data representations) are a complex and active research area in quantum computing. We provide detailed definitions, explanations and examples in Appendix A and B1 of our online technical report [4].

Bitwise Operation	Symbol	Description
AND	&	Bitwise AND of two operands
OR	|	Bitwise OR of two operands
NOT	~	Inverts bitstring of operand
Left Shift	<<	Shifts bits left by *n* bits.
Right Shift	>>	Shifts bits right by *n* bits.

Table 1: Bitwise Operations used in the SQL queries

with `H` *table via* `JOIN H ON H.in_s = (T0.s & 1)`. *The value of* `s` *of the output state is computed as* `(T0.s & ~1) | H.out_s`.

We can see from the above example that by using bitwise operators in Table 1, we can directly locate and manipulate individual qubits and directly access and operate on them.

Discussion. We have presented a declarative, SQL-based approach that specifies *what* operations gates perform without requiring users to manage *how* to implement them. Our work improves existing methods [2, 6] in the following aspects. First, Blacher et al. [2] encode gates as tensor computations, often requiring multiple columns per index dimension, leading to no clear performance advantage over NumPy, as shown in their experiments. By contrast, our integer-based encoding leverages bitwise operations, reducing storage overhead and potentially improving performance. [6] treats qubit states as strings, which increases storage costs and complicates indexing. Our approach includes CPU-native bitwise instructions and the integer indexing of qubit state, enabling more efficient lookups and compact storage than string-based representations in [6].

3 System Overview

Qymera is an end-to-end, web-based system demonstrating how RDBMSs are leveraged for quantum circuit simulation. As depicted in Fig. 1, it organizes its functionality across four different layers.

3.1 Circuit Layer

Existing frameworks like Qiskit and Cirq offer powerful APIs but lack intuitive graphical interfaces, limiting accessibility for non-programmers. Browser-based tools like Quirk[6] and the Quantum Länd Simulator[7] attempt to bridge this gap but have significant limitations: Quirk, built on JavaScript, supports only 16 qubits and lacks parameterized circuit support, while the Quantum Länd Simulator provides a QASM interface with limited and unintuitive handling of parameterized circuits.

To address these shortcomings, we introduce the *Circuit Layer*, shown in Fig. 3a, a flexible, user-friendly interface for quantum circuit construction and integration. It provides multiple methods for circuit input, including:

- **Graphical Circuit Builder:** An intuitive drag-and-drop interface that allows researchers, particularly those new to quantum computing, to visually construct quantum circuits.
- **File Upload:** Quantum researchers can upload circuits in standardized formats, such as JSON, to import pre-defined designs.
- **Parameterized Circuit Families:** Researchers can define parameterized circuits programmatically using various APIs.

[6]https://algassert.com/quirk
[7]https://thequantumlaend.de/quantum-circuit-designer/

3.2 Translation Layer

The *Translation Layer*, as detailed in Sec. 2, is responsible for converting abstract quantum circuits into executable SQL queries.

- **SQL Generation:** The translation layer parses the input circuit, identifies qubits and gates involved, and generates the corresponding SQL queries.
- **Gate Representation:** Quantum gates are stored as tables encoding input-output mappings and transition probabilities.
- **Query Optimization:** To improve performance, consecutive gates are fused into single SQL query where possible, minimizing intermediate results and leveraging database query optimizers.

3.3 Simulation Layer

The *Simulation Layer* handles the execution of the SQL queries generated in the Translation Layer. Quantum researchers can manage simulation runs as shown in Fig. 3b.

- **Backend Integration:** Qymera executes SQL queries on RDBMS engines to compute the resulting quantum state or measurement probabilities. It supports SQLite 2.6.0, and DuckDB 1.1.
- **Support for Multiple Methods:** Qymera allows quantum and database researchers to compare database-driven simulations with alternative simulation methods like tensor network and state-vector simulators (cuQuantum [1]) or decision diagram based simulators (MQT DD [9]).
- **Parameterized Simulations:** Quantum researchers can define families of circuits with varying parameters, and Qymera automates simulation across the parameter space.
- **Out-of-Core Simulation:** For large circuits that exceed memory capacity, Qymera leverages database features to efficiently manage intermediate states and I/O, enabling simulations at scales beyond traditional in-memory methods.

3.4 Output Layer

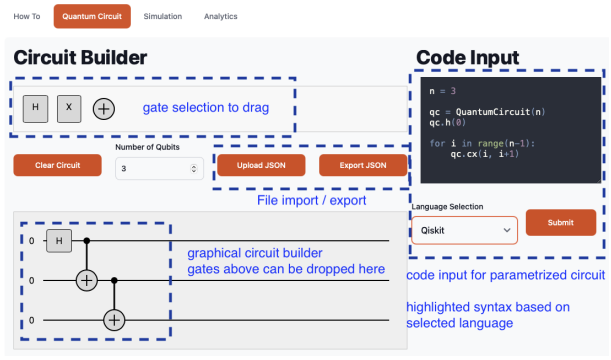
The *Output Layer*, as illustrated in Fig. 3c, provides researchers with tools to analyze simulation results and benchmarking metrics.

- **Simulation Results:** The system outputs the final quantum state, including measurement probabilities.
- **Performance Metrics:** Execution time, memory usage, and other relevant metrics are logged and displayed for each simulation method.
- **Visualization Tools:** Interactive plots enable the user to explore the behavior of circuits, compare simulation methods, and analyze performance across parameterized families of circuits.
- **Export and Reporting:** Results and visualizations can be exported for analysis or publication.

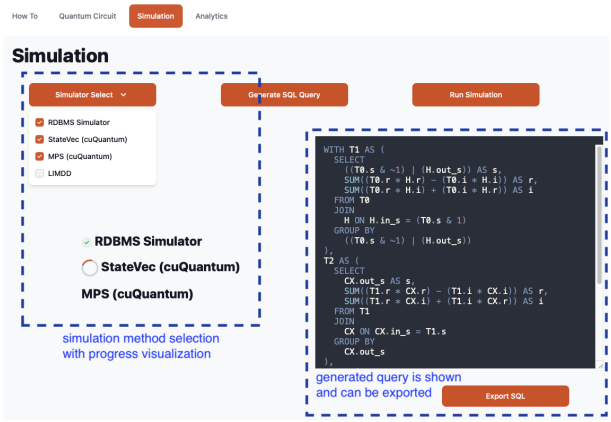
The Output Layer ensures that researchers derive insights from their simulations, through detailed analysis and high-level comparisons.

4 Demonstration Scenarios

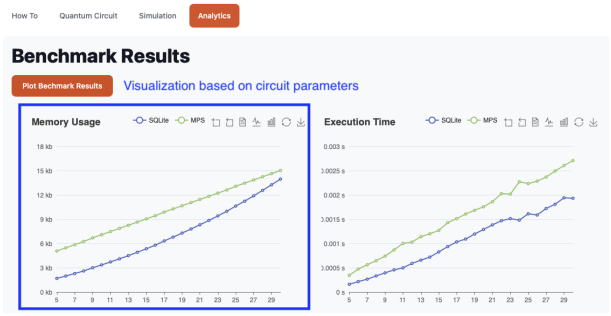
The demonstration of Qymera will showcase its application in quantum circuit simulation using RDBMSs, serving both as a simulation method and an infrastructural backbone for circuit exploration. SIGMOD attendees will interact with the system through an intuitive interface as shown in Fig. 3. They can explore Qymera's ability to convert quantum circuits into SQL queries, execute them efficiently



(a) Circuit Panel Tab



(b) Simulation Panel Tab



(c) Visualization Panel Tab

Figure 3: Qymera's core UI tabs.

within an RDBMS, and benchmark parameterized quantum circuits using various simulation backends.

The demonstration is structured into the following scenarios. **Quantum Algorithm Design and Testing.** This scenario highlights how Qymera enables rapid iteration and testing of quantum algorithms. For example, attendees will construct and analyze a simple quantum parity check algorithm, which determines whether the number of ones in a given bitstring is even or odd.

Qymera will translate the algorithm into SQL, execute it within an RDBMS, and allow attendees to inspect intermediate quantum states and measure execution performance. Additionally, attendees can compare the execution with other simulation techniques, such as state-vector simulation, to assess the trade-offs in performance and accuracy. This will demonstrate how researchers and developers can leverage Qymera to refine quantum algorithms efficiently and select the optimal simulation method for their needs. **Simulation Method Benchmarking.** Benchmarking different quantum circuit simulators is crucial for assessing performance trade-offs. In this scenario, attendees will evaluate the efficiency of various simulation backends using a GHZ state preparation circuit and an equal superposition of all possible states as test cases. The system will execute the circuit across different simulation approaches as introduced in Sec. 3.

Performance metrics, including execution time, and memory usage, will be analyzed, providing insights into the strengths and limitations of each simulator. This comparative analysis will highlight the conditions under which SQL-based simulation is advantageous and when alternative simulators may be preferable.

Educational Exploration of Quantum Computing Concepts. Qymera also serves as a tool to explore fundamental quantum computing principles. This scenario will focus on illustrating entanglement and superposition using the GHZ state as a case study. Attendees will construct a circuit, observe the evolution of quantum states through SQL queries, and explore measurement outcomes.

By interacting with Qymera, SIGMOD attendees will develop a deeper understanding of quantum mechanics concepts.

To further enhance educational effectiveness, Qymera will integrate interactive visualizations explicitly focused on quantum states and gate operations. Specifically quantum states visually represented as Block spheres and how they evolve upon applying different gates in the graphical circuit builder. This scenario is designed to provide a conceptual and interactive learning experience, making quantum computing principles more accessible.

Through the above scenarios, participants will gain a comprehensive understanding of Qymera, including its novel SQL-based approach to quantum circuit simulation, its role in benchmarking different simulation methods, and its educational potential.

Acknowledgments

This publication was supported (in part) by Dutch Research Council (VI.Veni.222.439).

References

[1] Harun Bayraktar et al. 2023. cuQuantum SDK: A High-Performance Library for Accelerating Quantum Science. arXiv:2308.01999 https://arxiv.org/abs/2308.01999
[2] Mark Blacher et al. 2023. Efficient and Portable Einstein Summation in SQL. *Proc. ACM Manag. Data* 1, 2, Article 121 (June 2023), 19 pages. doi:10.1145/3589266
[3] Rihan Hai, Shih-Han Hung, Tim Coopmans, Tim Littau, and Floris Geerts. 2025. Quantum Data Management in the NISQ Era. *PVLDB* 18, 6 (2025). to appear.
[4] Rihan Hai, Shih-Han Hung, Tim Coopmans, Tim Littau, and Floris Geerts. 2025. Quantum Data Management in the NISQ Era: Extended Version. https://arxiv.org/abs/2409.14111
[5] Lennart Schmidt et al. 2025. Rethinking MIMD-SIMD Interplay for Analytical Query Processing in In-Memory Database Engines. *CIDR* 12 (2025), 26.
[6] Immanuel Trummer. 2024. Towards Out-of-Core Simulators for Quantum Computing. In *Q-Data '24 workshop.*
[7] Xu Xiaosi et al. 2023. A Herculean task: Classical simulation of quantum computers. (2023). arXiv:2302.08880
[8] Kieran Young et al. 2023. Simulating Quantum Computations on Classical Machines: A Survey. (2023). arXiv:2311.16505
[9] Alwin Zulehner et al. 2019. How to Efficiently Handle Complex Values? Implementing Decision Diagrams for Quantum Computing. *ICCAD* (2019).