Quantum Storage Design for Tables in RDBMS

Tuodu Li
Zhejiang University
ltd16@zju.edu.cn

Gongsheng Yuan
Zhejiang University
Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
ygs@zju.edu.cn

Chang Yao
Zhejiang University
Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
changy@zju.edu.cn

Meng Shi
Zhejiang University
konicy@zju.edu.cn

Ziyue Wang
Zhejiang University
w.ziyue@zju.edu.cn

Ling Qian
China Mobile (Suzhou) Software Technology Co., Ltd
qianling@cmss.chinamobile.com

Jiaheng Lu
University of Helsinki
jiaheng.lu@helsinki.fi

ABSTRACT

Emergent quantum computing holds significant promise for achieving significant speedups in specific tasks by utilizing quantum phenomena, leading to increasing interest from researchers in incorporating quantum computing into their research fields. Considering that current database systems are struggling to store and process large datasets on classical computers, we could attempt to use quantum computers to handle big data, allowing for a significant reduction in storage requirements and speedups for a variety of database operations and analyses. However, to support relational tables of RDBMSs on quantum computers, relational data needs to be represented in a quantum-compatible format. In this paper, we propose two storage methods, *Quantum Column-oriented Store* (QCOS) and *Quantum Row-oriented Store* (QROS), tailored to store relational tables on universal quantum computers. We conduct theoretical analyses and simulation validations on the costs of qubit and quantum gates in those two storage methods. The results indicate that the qubit cost of both storage methods shows a logarithmic growth trend as the data quantity increases. Besides, both methods maintain linear requirements for *MCT* gates. We perform numerous experiments on various real quantum machines from IBM, and the results indicate that our approaches could enable existing devices to hold datasets.

VLDB Workshop Reference Format:
Tuodu Li, Gongsheng Yuan, Chang Yao, Meng Shi, Ziyue Wang, Ling Qian, and Jiaheng Lu. Quantum Storage Design for Tables in RDBMS. VLDB 2024 Workshop: The Second International Workshop on Quantum Data Science and Management (QDSM'24).

* Gongsheng Yuan and Chang Yao are corresponding authors.

1 INTRODUCTION

One core research topic in the field of Relational Database Management Systems (RDBMSs) concerns harnessing cutting-edge hardware infrastructures to manage data. The recent advent of universal quantum computers [1, 2], especially gate-based quantum computers, heralds disruptive changes in the architectures supporting relational data organization and retrieval fundamentally and offers exciting opportunities for optimizing the processing of relational data with quantum technologies.

In this paper, we aim to design two quantum storage formats for relational tables in RDBMSs, offering two distinct advantages. Firstly, with the properties of superposition states, a minuscule number of qubits can store a vast amount of data, which means a novel solution for storing big data. That is, such quantum advantages may delineate a path toward reducing storage requirements for RDBMSs. Secondly, these quantum architectures facilitate the seamless conversion of relational tables into quantum states. This intrinsic mapping naturally allows native quantum algorithms to accelerate critical relational operations such as selections, joins, projections, and other database tasks. By quantizing relational tables, one could hypothesize the prospect of exponential speedups for a variety of database operations and analyses. This integration of quantum information processing with relational databases harbors transformative implications.

Although there are several prior works providing potential formats for storing data on quantum circuits, those studies have not presented a specific quantum storage scheme for database systems, nor have they offered corresponding cost and performance analyses. In this paper, we propose two distinct approaches for storing relational tables on a universal quantum computer, namely *Quantum Column-oriented Store* (QCOS) and *Quantum Row-oriented Store* (QROS), akin to the column-oriented store [23] and row-oriented store [6] paradigms of RDBMSs on classical computers. To illuminate the characteristics of each approach, we theoretically quantify their requirements of quantum resources, focusing analysis on primary cost metrics such as the number of qubits and quantum

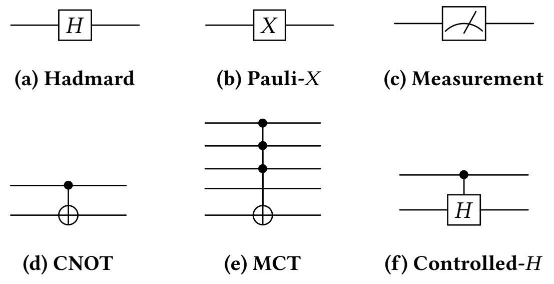


Figure 1: Six basic quantum gates

gates (mainly *MCT* gates). To verify our analysis, we perform extensive numerical simulations to observe the cost of QCOS and QROS circuits. To estimate practical performance under realistic constraints, we implement our proposed quantum tabular storage formats on IBM's openly accessible quantum computing simulators and currently available physical hardware, respectively. We hope the proposed quantum storage paradigms for relational data and corresponding experimental evaluations deliver valuable insights toward realizing fully-fledged quantum database systems.

In detail, our contributions include:

(1) We propose two fundamentally distinct methods (i.e., QCOS and QROS) for storing relational tables on universal quantum computers.
(2) We analyze the qubit and *MCT* gate costs for these two storage methods and verify the conclusions through numerical simulations.
(3) We deploy two storage methods on IBM's simulators and real quantum computers, respectively, to examine the performance of our two quantum storage methods in both ideal and real-world environments.

Related work In recent years, more and more researchers have been dedicated to developing quantum algorithms to address complex challenges in the field of databases [3, 4, 7, 9, 10, 14, 16, 17, 19–22, 24, 26, 29, 32]. As quantum hardware advances rapidly, the necessity of designing standardized data models for database storage on quantum hardware becomes increasingly evident [31]. While several preliminary solutions or prototypes [5, 7, 11, 12, 18, 27, 30] have been proposed, these efforts simply treat quantum storage of relational tables as a middle procedure for application algorithms like quantum search. There is no paper offering a complete and viable method for the quantum storage of relational tables, coupled with an analysis and validation of the quantum resource cost it entails. Furthermore, to the best of our knowledge, no prior work has conducted quantum storage of relational tables on real quantum computers.

2 PRELIMINARY

In this section, we concisely introduce some basic knowledge of quantum computing related to our research content [13, 15].

2.1 Qubit

Qubit is the basic unit of quantum computing. Resembling the bits in a classical computer, the states of a qubit can be represented by 0 and 1 after measurement(**Measurement** will be introduced later in the Quantum Gate subsection). But the difference is that when a qubit is not measured, as a quantum system, it is in a superposition state of 0 and 1. If a qubit is measured, the result may be 0 or 1. After the measurement, the state of the qubit will remain unchanged and certain.

2.2 State Vector

The qubit is the fundamental information carrier in a quantum computing system. Similarly to classical bits, the computational basic states of a qubit can be expressed as $|0\rangle$ and $|1\rangle$ (denoted in Dirac notation), which can also be represented as state vectors $[1\ 0]^T$ and $[0\ 1]^T$, respectively, in a Hilbert space. A salient distinction, however, is that before measurement, a qubit state is described by a coherent superposition of these two basic states according to the postulates of quantum mechanics. That is, the superposition state of a single qubit can be mathematically represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex probability amplitudes such that $|\alpha|^2 + |\beta|^2 = 1$. Upon measurement, the qubit state probabilistically collapses to either $|0\rangle$ or $|1\rangle$, with probabilities $|\alpha|^2$ and $|\beta|^2$, respectively.

The state of multiple qubits can be elegantly expressed as: $|\psi_n\rangle = \sum_{i_1,i_2,\ldots,i_n \in \{0,1\}} a_{i_1 i_2 \ldots i_n} |i_1 i_2 \ldots i_n\rangle$, where $|i_1 i_2 \ldots i_n\rangle$ is the basic vector of multi-qubit space. That is, before measurement, the state of a quantum circuit comprising multiple qubits can be described as a quantum superposition over the computational basic states. The orthonormal basic states of multi-qubit systems, along with their associated probability amplitudes, provide an ideal probabilistic representation for encoding data amenable to quantum algorithms. Critically, the number of basic states allowed within the Hilbert space grows exponentially with the number of qubits, endowing quantum computing with exponential potential for enhanced performance over classical analogues.

2.3 Quantum Gate

Quantum gates, which can manipulate the superposition state of qubits, serve as the fundamental building blocks of quantum circuits. They can be divided into two types: single-qubit gates and multi-qubit gates.

In this paper, the single-qubit gates used to implement the proposed storage design are the Hadamard gate (denoted as *H*) and the Pauli-*X* gate (denoted as *X*), as shown in Figure 1. The Pauli-*X* gate is similar to the NOT operation in classical computing, which can turn $|0\rangle$ into $|1\rangle$ or $|1\rangle$ into $|0\rangle$. The *H* gate will turn $|0\rangle$ or $|1\rangle$ into a uniform superposition state. Their mathematical descriptions are as follows:

$$X(a|0\rangle + b|1\rangle) = a|1\rangle + b|0\rangle$$

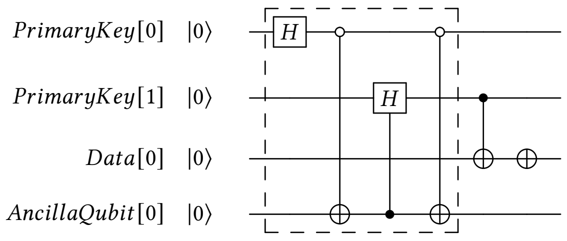
$$H(a|0\rangle + b|1\rangle) = a\frac{|0\rangle + |1\rangle}{\sqrt{2}} + b\frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Besides, this article involves the application of multi-qubit gates, including *CNOT* gate, *MCT* gate, and Controlled-*H* gate. The *CNOT* gate applies the Pauli-*X* gate on the target qubit only when the control qubit is in the $|1\rangle$ state, otherwise leaving the target unchanged. The mathematical description of the *CNOT* gate is as follows:

$$CNOT(a|10\rangle + b|11\rangle + c|00\rangle + d|01\rangle) = a|11\rangle + b|10\rangle + c|00\rangle + d|01\rangle$$

Primary Key	*Data*
0	1
1	1
2	0

(a) A relational table


$PrimaryKey[0]$ $|0\rangle$
$PrimaryKey[1]$ $|0\rangle$
$Data[0]$ $|0\rangle$
$AncillaQubit[0]$ $|0\rangle$
H
H

(b) A quantum circuit

Figure 2: An example of transforming a relational table into a quantum circuit. The top qubit represents the least significant qubit, and the bottom qubit represents the most significant qubit.

In addition to the *CNOT* gate described previously, this work also utilizes the class of generalized Toffoli gates with multiple controls (i.e., Multiple-Control Toffoli gate or *MCT* gate) as well as the Controlled-H gate (i.e., *CH* gate). Specifically, the *MCT* gate applies X to the target qubit conditioned on all control qubits being in $|1\rangle$. The *CH* gate performs H on the target qubit depending on the control qubit being $|1\rangle$. Finally, as illustrated in Figure 1, *Measurement* gate allows us to observe a qubit.

3 DATA STORAGE ON QUANTUM CIRCUIT

Table 1 shows the symbols required for introducing our quantum storage methods and cost analysis. Among them, *ar*, *ac*, and *ad* are three indicators whose values can reflect the overall characteristics of r_i, c_i, and d_{ij}. Although they may not be equivalent to the average values, they all represent a value located near the middle of the data distribution.

We start our work with a common method to transform a relational table into a quantum circuit [12, 28], which involves two components: one is the construction of quantum basic states, and the other is the storage of records. Based on this method, we propose two storage schemas for relational tables: QCOS and QROS.

As shown in Figure 2, to obtain the quantum representation of the relational table in Figure 2(a), we need to construct a quantum circuit whose quantum state serves as a representation of Figure 2(a). Firstly, using the circuit within the dotted box in the Figure 2(b), we prepare a quantum state (i.e., $\frac{1}{2}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle + \frac{1}{2}|10\rangle$) for the qubits stored in *PrimaryKey* register (including two qubits: $PrimaryKey[0]$ and $PrimaryKey[1]$) and use this state to represent *Primary Key* of the relational table. Secondly, the value of *PrimaryKey* register, regarded as the controlling condition of *MCT* gates, is used to set the value of *Data* register (including one qubit: $Data[0]$). Please note that *MCT* gates used here are optimized by [28]. Finally, the quantum state of the constructed quantum circuit ($|Data[0]Primarykey[1]Primarykey[0]\rangle$) in Figure 2(b) is: $\frac{1}{2}|100\rangle + \frac{1}{\sqrt{2}}|101\rangle + \frac{1}{2}|010\rangle$.

Table 1: Symbol Description

Symbol	Description
n	# of relational tables
r_i	# of rows in ith relational tables
c_i	# of columns in ith relational tables
d_{ij}	# of **bits** for one element in the jth column of the ith relational table(e.g., for one element of type *char* in ith table and jth column, $d_{ij} = 8$)
dq	$dq = \sum_{i=1}^{n} (r_i * \sum_{j=1}^{c_i} d_{ij})$ (i.e., number of **bits** for tables of a database, namely **data quantity** also.)
mr	$\max_{i\in(1,n)} (r_i)$
mc	$\max_{i\in(1,n)} (c_i)$
md	$\max_{i\in(1,n),j\in(1,c_i)} (d_{ij})$
ar	$ar^n = \prod_{i=1}^{n} r_i$, ar is a value between the maximum and minimum of r_i, and related with the database schema.
ad_i	$ad_i = \frac{\sum_{j=1}^{c_i} d_{ij}}{c_i}$, i.e. ad_i is the average number of d_{ij} of ith table
ad	$ac * ad * n = \sum_{i=1}^{n} ad_i \times c_i$
ac	Although *ad* and *ac* can not be calculated by the equation, it is obvious that there exist cases where both are values between the maximums and minimums of ad_i and *ac*. And they are related with the schema of the database.
CQ	Qubits cost
CM	*MCT* Gates cost

3.1 Quantum Row-oriented Store

QROS is composed of multiple quantum circuits (each quantum circuit represents a relational table). Based on the method described above, one relational table can be transformed to a quantum circuit. And for several tables from one database, their corresponding quantum circuits together form QROS circuit. The following format is designed to store relational tables in the quantum circuit:

$$|Table_{n-1}\rangle ... |Table_1\rangle |Table_0\rangle$$

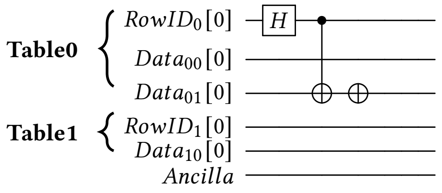


Figure 3: Quantum circuit for QROS. The top qubit represents the least significant bit, and the bottom qubit represents the most significant bit.

The required number of qubits for each register in the QROS circuit is shown in Table 2. For example, to store **Table0** in Figure 4(a) on the quantum circuit of Figure 3, we need to apply $f(2) = 1$, 1, and 1 qubits for $RowID_0$, $Data_{00}$ and $Data_{01}$ registers, respectively. Next, an H gate is applied to the qubit $RowID_0[0]$ for the purpose of preparing the state in $RowID_0$ register. Then, the optimized MCT gates are applied to $Data_{00}[0]$ and $Data_{01}[0]$ qubits conditioned on the state of $RowID_0[0]$. Now, we have constructed a quantum circuit for **Table0** in the way of QROS. Similarly, we could construct a quantum circuit for **Table1**. After combining them together, we could obtain quantum circuit in Figure 3.

In the definition of QROS, it is evident that a basic state in registers of $Table_i$ represents a row of $Table_i$. In the QROS quantum circuit, a basic state is composed of n rows, where each row corresponds to a unique record of different relational tables. Since this, we call this storage structure *Quantum Row-oriented Store*.

Table 2: The Required Number of Qubits in Each Quantum Registers for QROS

Register	# of Qubits
$RowID_i$	$f(r_i)$
$Data_{ij}$	d_{ij}
$Ancilla$	1

3.2 Quantum Column-oriented Store

QCOS does not directly store the original relational tables. It stores a table (called **QuadrupleTable**) containing information from multiple original relational tables. To store relational tables on the quantum circuit, we represent multiple tables with our proposed *quadruple table*, a table with only four columns, i.e., **QuadrupleTable**(*tableID*, *columnID*, *rowID*, *data*). Subsequently, we could keep this quadruple table on the quantum circuit in the way of QCOS.

Table 3: The Required Number of Qubits in Each Quantum Register for QCOS

Register	# of Qubits
$TableID$	$f(n)$
$ColumnID$	$f(mc)$
$RowID$	$f(mr)$
$Data$	md
$Ancilla$	1

Table0

rowID	*column0*	*column1*
0	0	1
1	0	0

Table1

rowID	*column0*
0	0

(a) Multiple Tables

QuadrupleTable

tableID	*columnID*	*rowID*	*data*
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0

(b) A Quadruple Table

Figure 4: Mapping multiple relational tables into one quadruple table

As shown in Figure 4, given two tables (**Table0** and **Table1** in Figure 4 (a)), we map them into our designed quadruple table, i.e., Figure 4 (b). For example, the value in the first row and in the second column of **Table0** is 0. To map it into **QuadrupleTable**, we set $tableID = 0$, $columnID = 0$, $rowID = 0$, and $data = 0$. This is, we preserve it as a tuple in **QuadrupleTable**.

After getting **QuadrupleTable**, it is essential to create a quantum circuit structure corresponding to the **QuadrupleTable** schema. Consequently, the following format is our proposed quantum relational schema for relational tables, which requires five quantum registers: a $TableID$ register, a $ColumnID$ register, a $RowID$ register, a $Data$ register, and an $Ancilla$ register.

$$|Ancilla\rangle |Data\rangle |RowID\rangle |ColumnID\rangle |TableID\rangle.$$

To obtain the necessary amount of qubits for the above quantum registers, we define a function f:

$$f(x) = \max(\lceil log_2(x) \rceil, 1), x \in N^+ \qquad (1)$$

Note that each quantum register must minimally comprise a single quantum bit, and the definition of f can ensure this when $x = 1$. With the function $f(x)$, we could get the required number of qubits in each quantum register for QCOS in Table 3.

Since the primary key of **QuadrupleTable** is comprised of *tableID*, *columnID*, and *rowID*, in order to prepare the value of the primary key, the first step is to prepare the quantum state of *tableID* register, and the second step is to use the state of *tableID* register as conditions to prepare the states of *columnID* and *rowID* registers. And finally, the state of *data* register can be set by *tableID*, *columnID*, and *rowID* registers.

For instance, if we want to store the **QuadrupleTable** (Figure 4 (b)) in the previous five registers, we need to apply for $f(2) = 1$ qubit for $TableID$ register. Similarly, the required number of qubits for *ColumnID*, *RowID*, and *Data* are 1, 1, and 1, respectively (see Figure 5). Next, we apply the H gate to the qubit $TableID[0]$. Based on the superposition state of $TableID[0]$ (i.e., $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$), we prepare the states of *ColumnID* and *RowID* registers. Lastly, the *MCT* gates are applied to *Data* register conditioned on the value of the primary key. Since each basic state in the QCOS circuit represents a column data (i.e., one record in **QuadrupleTable**), we call this storage structure as *Quantum Column-oriented Store*.

4 COST ANALYSIS

The cost of QCOS and QROS is mainly divided into two parts: the number of qubits and the number of *MCT* gates. In this section, we show the Qubits cost and *MCT* gate cost of QCOS and QROS circuits.

4.1 Qubit Cost

Table 3 and Table 2 show the required number of qubits for the quantum registers of QCOS and QROS circuits. Naturally, the total number of qubits for QCOS and QROS is the sum of the quantities of qubits in each quantum register. Based on Table 3 and Table 2, the total number can be obtained as follows:

$$CQ_{QCOS} = f(n) + f(mc) + f(mr) + md + 1$$

$$CQ_{QROS} = \sum_{i=0}^{n-1} (f(r_i) + \sum_{j=0}^{c_i - 1} d_{ij}) + 1$$

4.1.1 *Qubit Cost of QCOS.*

In the QCOS circuit, generally, the number of qubits is logarithmic to most schema parameters, such as *n*, *mr*, and *mc*. Although the qubit quantity is linear with the *md*, considering QCOS needs only one data register and the qubits count of it will not change as the number of tables escalates and the dimensions of these tables expand, the amount of qubits in a QCOS circuit is still roughly logarithmic to data quantity.

4.1.2 *Qubit Cost of QROS.*

Combining with the definition of f, in most cases, CQ_{QROS} can be expressed as:

$$\sum_{i=0}^{n-1} (\lceil log_2(r_i) \rceil + ad_i \times c_i)$$

The ad_i is the average of d_{ij} for all columns in ith table.

Continuing to use average values to convert accumulation into multiplication, we can get CQ_{QROS} as:

$$\sum_{i=0}^{n-1} (\lceil log_2(r_i) \rceil) + ad \times ac \times n$$

If we define:

$$\lceil log_2(r_i) \rceil = \alpha_i \times log_2(r_i) \; (\alpha_i \in [1,2])$$

Then we can get:

$$\sum_{i=0}^{n-1} (\lceil log_2(r_i) \rceil) = \left(\prod_{i=0}^{n-1} \alpha_i\right) log_2 \left(\prod_{i=0}^{n-1} r_i\right)$$

Obviously, $\prod_{i=0}^{n-1} r_i$ can be expressed as:ar^n
And if we define:$\prod_{i=0}^{n-1} \alpha_i = \alpha$, we get:

$$\sum_{i=0}^{n-1} (\lceil log_2(e_i) \rceil) = \alpha \times n \times log_2(ar)$$

To sum up, the number of qubits for QROS circuit is:

$$CQ_{QROS} = \alpha \times n \times log_2(ar) + ad \times ac \times n$$

Please note that *ar* is just a value between the maximum and minimum values of r_i, not an average in the mathematical sense. However, it is still related to the number of rows in each table. From the above formula, it is obvious that the number of qubits in the QROS circuit has a linear relationship with *n*, and the value distributions of c_i and d_{ij}(As said in Table 1, *ad* and *ac* are values between the maximum and minimum of d_{ij} and c_i), and a logarithmic relationship with *ar*.

4.2 MCT Gate Cost

Although the quantity of *MCT* gates is influenced by the value of the primary key and data [28], it remains feasible to analyze the mathematical expectation of the number of *MCT* gates under a given circumstance.

Given *nc* qubits acting as candidate control qubits, implying a range of 0 to *nc* control qubits for the *MCT* gates, total potential occurrences of *MCT* gates in this circuit can be expressed as:

$$N = \sum_{i=0}^{nc} C_{nc}^{i} = 2^{nc}$$

The *MCT* gate that appears in the final circuit optimized by the method mentioned in [28] must be a combination of these N gates, and each gate appears at most once because two identical *MCT* gates will cancel out. So the number of possible sets of *MCT* gates in final circuit is: $\sum_{i=0}^{N} C_N^i = 2^N$. And the probability of the set with M *MCT* gates is: $\frac{C_N^M}{2^N}$. So the mathematical expectation of M is:

$$E(M) = \frac{\sum_{m=0}^{N} mC_N^m}{2^N} = \frac{2^{N-1} \times N}{2^N} = 2^{nc-1}$$

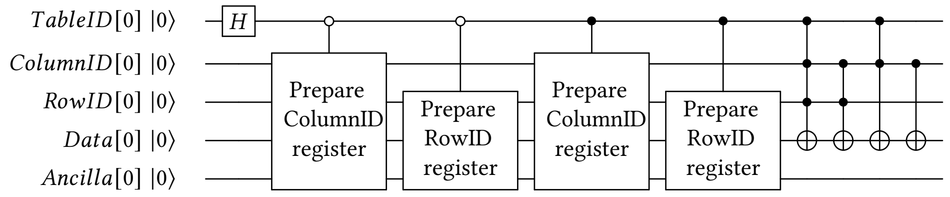


Figure 5: Quantum circuit for QCOS. The top qubit represents the least significant bit, and the bottom qubit represents the most significant bit. Note that *Prepare ColumnID register* only apply MCT **gates to** $ColumnID$ **register, and *Prepare RowID register* only apply** MCT **gates to** $RowID$ **register. And both of them use** $Ancilla$ **register as controlled qubit of** MCT **gates**

4.2.1 *MCT Gate Cost of QCOS.* In the QCOS circuit, the nc control qubits are composed of $TableID$ register, $ColumnID$ register, and $RowID$ register. As shown before, in the QCOS circuit, $nc = CQ_{QCOS} - md$. So combining the definition of f, we can get the number of MCT gates for one target qubit:

$$O\left(2^{\lceil log_2(n)\rceil} \times 2^{\lceil log_2(mr)\rceil} \times 2^{\lceil log_2(mc)\rceil}\right)$$

Similar to QROS, the formula above can be written as:

$$O\left(\beta \times \gamma \times \eta \times n \times mr \times mc\right)$$

The β, γ and η are analogous to α.

The best case is when all tables share an identical number of rows and columns, and in that case, the number of MCT gates is:

$$O\left(\beta \times \gamma \times \eta \times dq \times md\right)$$

And the worst case is when the data quantity of most tables is much smaller than the largest table, and then the number of MCT gates is:

$$O\left(\beta \times \gamma \times \eta \times n \times dq \times md\right)$$

4.2.2 *MCT Gate Cost of QROS.* In the ith table of the QROS circuit, n' is the number of qubits in $RowID_i$ register, and in most cases $nc_i = \lceil log_2\,(r_i)\rceil$. So $2^{nc_i - 1}$, in big O notation, can be denoted as $O(2^{\lceil log_2(r_i)\rceil})$. At the same time, for a table, there are $\sum_{j=0}^{c_i-1} d_{ij}$ target qubits, i.e., qubits in the data registers. So, for the total QROS circuit, the number of MCT gates is:

$$O(\sum_{i=0}^{n-1} 2^{\lceil log_2(r_i)\rceil} \times c_i \times ad_i)$$

.

The ad_i , in the ith table, is a number between the maximum of d_{ij} and the minimum, which can satisfy:

$$\sum_{j=0}^{c_i-1} d_{ij} = c_i \times ad_i$$

It is readily inferred through mathematical deduction that:

$$2^{\lceil log_2(r_i)\rceil} = \alpha_i \times r_i\ (\alpha_i \in [1,2])$$

Then, the number of MCT gates for a QROS circuit can be denoted as:

$$O(\sum_{i=0}^{n-1} \alpha_i \times r_i \times c_i \times ad_i)$$

And if we define:

$$dq = \sum_{i=0}^{n-1} r_i \times c_i$$

The number of MCT gates in a QROS circuit is:

$$O(dq \times ad \times \alpha)$$

The ad and α satisfy:

$$dq \times ad \times \alpha =$$
$$\sum_{i=0}^{n-1} e_i \times c_i \times ad_i \times \alpha_i$$

Please note that in both QROS and QCOS, because of the definition of f, The impact of r_i, n, mr, and mc on the number of MCT gates has stage characteristics. For example, when $mr \in (9, 16]$, the value of $\lceil log_2(mr)\rceil$ is always 4.

5 EXPERIMENTAL EVALUATION

In this section, we begin by conducting the simulations of the qubit and MCT gate cost; next, we implement our quantum storage methods on qiskit simulators (i.e., noiseless and noisy simulators) and real quantum computers to verify their performances.

5.1 Experimental Setup

Datasets. For cost simulation, we randomly generate datasets so that the dq of these datasets increases within a range, and then calculate the number of qubits and MCT gates required for these datasets. To store data on simulators and real devices, considering the limitations of the current hardware, we designed some very simple datasets with only 0 and 1 values. We use the form $(r_0 \times c_0, r_1 \times c_1, ..., r_{n-1} \times c_{n-1})$ to represent them.

Metrics. For cost simulation, the metrics of cost are the number of qubits and the number of MCT gates. And we use **Data Quantity** (i.e., dq) to evaluate the size of the dataset. To store data on simulators and real devices, we introduce **Valid Ratio** to evaluate the performance of the storage. **Valid Ratio** means the proportion of correct records of measurement results on quantum circuits. To get **Valid Ratio**, the state of QCOS and QROS circuits are prepared repeatedly, and measurements are made on circuits. The measurement results are then converted into the form of relational table records. If the converted relational table records exist in the original relational tables, the measurement results are called **correct**

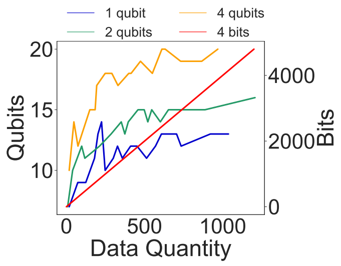


Figure 6: QCOS Qubits. The red curve represents the number of bits required to store relational tables on a classical computer. The number of bits is on the right y-axis. The other three curves represent the number of qubits required when the same tables are stored using the QCOS method. The number of qubits is on the left y-axis. The legend illustrates the number of bits or qubits required for each data in the relational tables corresponding to each curve.

measurement results. The formula is as follows:

$$Valid\ Ratio = \frac{\#\ of\ correct\ measurement\ results}{\#\ of\ measurements}$$

5.2 Cost Simulation

In this subsection, building upon the preceding theoretical analysis, we will conduct numerical simulation experiments centered on the schema parameters influencing the costs of QCOS and QROS to validate our theoretical findings.

5.2.1 Qubit cost. Figure 6 depicts the qubit cost in QCOS. By regulating the number and dimensions of tables within a defined range, we observe the qubit cost across various **Data Quantity**. It is evident that when **Data Quantity** reaches a specific value, the trend in qubit cost exhibits a distinct logarithmic pattern. In addition, for the same **Data Quantity**, the bit cost of classical storage is much higher than the qubit cost of QCOS.

Figure 7 illustrates the variation in the number of qubits within the QROS circuit across different schema parameters. When holding other schema parameters constant and only modifying *ar*, the alteration in the number of qubits exhibits distinct phased and logarithmic characteristics. Conversely, adjustments in *ac* result in an evident linear trend in the number of qubits. Furthermore, increases in the number of tables and qubits required of data types do not influence the trend of qubit count variation(still logarithmic or linear); however, they do lead to an increase in qubit cost under equivalent **Data Quantity**. Classical storage has far higher bit costs than QROS in all cases.

5.2.2 MCT Gate Cost. For QCOS circuits, our theoretical analysis indicates that the impact of schema parameters varies with changes in the overall schema structure. As illustrated in Figure 8, when comparing the best case (where all tables are of identical size) and the worst case (with one larger table while the others have only one row and one column), it is observed that, under the same **Data Quantity**, the best case incurs significantly lower *MCT* gate cost compared to the worst one. Moreover, in the latter case, an increase in the number of tables leads to a proportional increase in cost. In

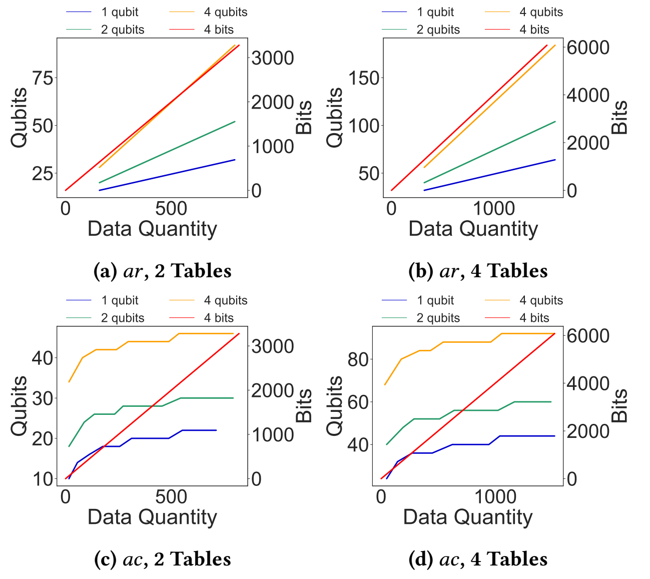


Figure 7: QROS Qubits. *ar* or *ac* means that other schema parameters are constant, and the *ar* or *ac* is modified. 2 or 4 Tables mean that the number of tables is 2 or 4. The red curve represents the number of bits required to store relational tables on a classical computer. The number of bits is on the right y-axis. The other three curves represent the number of qubits required when the same relational tables are stored using the QCOS method. The number of qubits is on the left y-axis. The legend illustrates the number of bits or qubits required for each data in the relational tables corresponding to each curve.

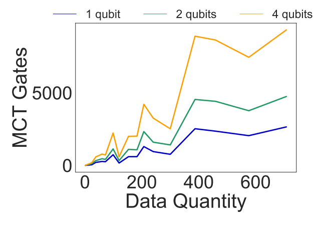
random cases, where the number and dimensions of tables assume random values within a specified range, QCOS exhibits significant fluctuations and abrupt increases in *MCT* gate consumption. This aligns with the phased characteristics outlined in our theoretical analysis.

The *MCT* gate cost for QROS exhibits a simpler behavior than that of QCOS. This is evident in Figure 9, where the *MCT* gate cost for QROS generally demonstrates a linear increase with the growing **Data Quantity**. Given that the schema parameters are subject to randomness, they exhibit fluctuations over time.

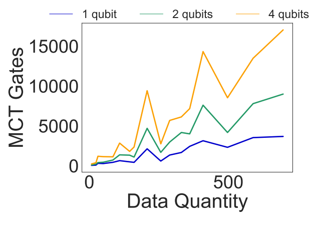
5.3 Storage on Simulators and Real Devices

Table 4: Dataset Description. (2×2) **means there is only one relational table with 2 rows and 2 columns in this dataset.** $(1 \times 2, 2 \times 1)$ **means that this dataset includes two tables. The size of the first table is one row and two columns, and the size of the second table is two rows and one column.**

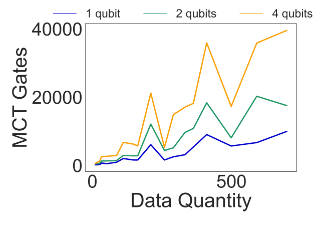
Symbol	Description
dataset_0	(2×2)
dataset_3	$(1 \times 2, 2 \times 1)$



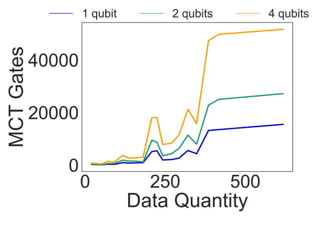
(a) Best Case



(b) Worst Case, 2 tables



(c) Worst Case, 4 tables



(d) Rondom Case

Figure 8: QCOS MCT Gates. The best case means that tables in the dataset hold the same number of rows and columns. The worst case means that except for one table that has multiple rows and columns, the other tables have only one row and one column. Random case means that the numbers of rows and columns for tables are generated randomly. 2 or 4 Tables means that the number of tables is 2 or 4. The legend illustrates the number of qubits required for each data in the relational tables corresponding to each curve.

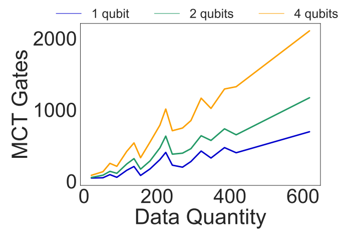


Figure 9: QROS MCT Gates. The legend illustrates the number of qubits required for each data in the relational tables corresponding to each curve.

Table 4 shows two datasets we use in this subsection. (2×2) means there is only one relational table with 2 rows and 2 columns in this dataset. $(1 \times 2, 2 \times 1)$ means that this dataset includes two tables. The size of the first table is one row and two columns, and the size of the second table is two rows and one column. More datasets and experiments can be found in **appendix A**.

To explore the performance of storing relational tables in simulators and existing universal quantum computers, we execute experiments on two datasets. The results are shown in Figure 10. Using the QROS method, the **Valid Ratios** of real quantum devices are consistent with the noisy simulator on the simple dataset. But when the size of the dataset set grows, the **Valid Ratios** of real quantum devices are almost 0. Although the **Valid Ratios** obtained by running the QCOS circuit on real devices are more stable than QROS, it is still far behind the noisy simulator. This means that existing quantum computers not only have errors in quantum gates but also have other unstable factors, which will further reduce the accuracy of relational table storage, especially when the relational table is large and complex.

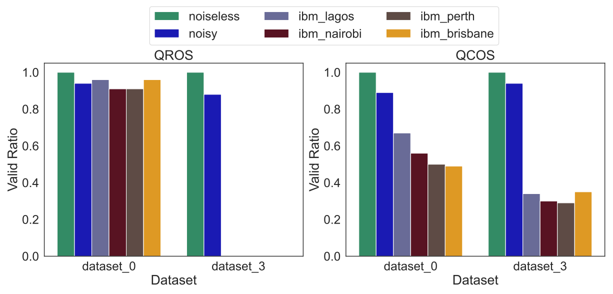


Figure 10: Results on simulators and real devices. The results of real devices on the latter dataset are invisible in the left picture because the Valid Ratios are almost 0. To calculate Valid Ratios, states are repeatedly prepared and measured, and the measurement results are compared with records in relational tables.

6 DISCUSSION

The difference between real devices and simulators. Experimental results show that there is still a huge gap between the simulator results and the real devices. This may be because current devices still have technical limitations, such as quantum decoherence and measurement errors. However, the results of the simulators show that if the real hardware can control errors within a certain range, our method can store relational tables effectively. This opens the way to using quantum circuits to process relational tables.

QCOS vs. QROS. From a cost perspective, the *MCT* gate costs of QCOS and QROS both increase linearly, but the qubit cost of QCOS increases logarithmically, so QCOS seems to be a more resource-saving storage method. In terms of accuracy, there is not much difference between QCOS and QROS on the simulators, but on real devices, QCOS's performance is relatively stable. Therefore, considering only the cost and effect of storage, QCOS is better than QROS. However, considering the complex circuit design of QCOS, QROS may be a simpler way to implement relational table storage.

Errors in quantum storage. Quantum gate errors are inevitable in existing quantum circuits. However, considering that classical database management systems often need to verify query results because of some errors, the verification process can also be added to the quantum storage of relational tables [8].

Inevitable Superposition State Destroying. It is inevitable that measuring quantum circuits representing relational tables will lead to superposition state destruction. In order to improve the utilization of quantum superposition states, several operations that require measurement (such as several query operations) can be compressed together firstly, and then the results of these operations can be obtained through one measurement. In addition, it is possible to specifically perform operations on quantum circuits for which quantum algorithms have advantages (that is, some operations that cannot be processed well by classical computers). It is also possible to learn from the *QRAM*[25] architecture to alternately perform unitary transformation and measurement to prevent blocking caused by superposition state destroying.

Relational Operations on Quantum circuits. Considering that gate-based quantum computers are a form of Turing machines, all relational operations can theoretically be implemented by quantum gate circuits. Specific to our circuit design, for example, we can use the quantum search algorithm to implement the join operation on *QROS* circuit. The basic idea is constructing an oracle(using $XNOR$ gate) to judge whether multiple data registers from multiple tables are the same. For example, a basic join statement like this: *SELECT t1.c1 FROM t1 JOIN t2 ON t2.c2 = t1.c1*, its oracle can be constructed by using $XNOR$ gate on corresponding two data registers. Such a quantum join operation will benefit from the advantages of the quantum search algorithm (i.e., the time complexity is $O(log_2(n))$), achieving acceleration over classical databases. We leave it as one of our future works.

7 CONCLUSION

In this paper, we present two novel techniques, QCOS and QROS, for mapping relational tables of RDBMSs into quantum states on universal quantum computers. Our cost simulations show that both methods can use limited qubits to keep large datasets. Besides, the MCT gate cost changes linearly with the data quantity. Finally, based on experimental results on simulators and real devices, our method can convert relational tables to quantum state well in an ideal environment or with some noise. In summary, our findings offer valuable insights and serve as essential references for developing quantum storage methods of databases.

ACKNOWLEDGMENTS

We would also like to thank all the reviewers for their valuable comments and helpful suggestions. This work is supported in part by the Zhejiang Provincial Natural Science Foundation (LD24F020015).

REFERENCES

[1] 2023. IBM's roadmap for scaling quantum technology. https://research.ibm.com/blog/ibm-quantum-roadmap.

[2] 2023. IonQ Forte. https://ionq.com/quantum-systems/forte.

[3] Tim Bittner and Sven Groppe. 2020. Avoiding blocking by scheduling transactions using quantum annealing. In *Proceedings of the 24th Symposium on International Database Engineering & Applications*. 1–10.

[4] Tim Bittner and Sven Groppe. 2020. Hardware accelerating the optimization of transaction schedules via quantum annealing by avoiding blocking. *Open Journal of Cloud Computing (OJCC)* 7, 1 (2020), 1–21.

[5] Paul Cockshott. 1997. Quantum relational databases. *arXiv preprint quant-ph/9712025* (1997).

[6] Edgar F Codd. 1970. A relational model of data for large shared data banks. *Commun. ACM* 13, 6 (1970), 377–387.

[7] Tobias Fankhauser, Marc E Solèr, Rudolf M Füchslin, and Kurt Stockinger. 2021. Multiple query optimization using a hybrid approach of classical and quantum computing. *arXiv preprint arXiv:2107.10508* (2021).

[8] Alexandru Gheorghiu, Theodoros Kapourniotis, and Elham Kashefi. 2019. Verification of quantum computation: An overview of existing approaches. *Theory of computing systems* 63 (2019), 715–808.

[9] Sven Groppe and Jinghua Groppe. 2021. Optimizing Transaction Schedules on Universal Quantum Computers via Code Generation for Grover's Search Algorithm. In *Proceedings of the 25th International Database Engineering & Applications Symposium*. 149–156.

[10] Le Gruenwald, Tobias Winker, Umut Çalikyilmaz, Jinghua Groppe, and Sven Groppe. 2023. Index Tuning with Machine Learning on Quantum Computers for Large-Scale Database Applications.. In *VLDB Workshops*.

[11] Amor Gueddana, Rihab Chatta, and Moez Attia. 2014. CNOT-based design and query management in quantum relational databases. *International Journal of Quantum Information* 12, 04 (2014), 1450023.

[12] Amor Gueddana, Rihab Chatta, and Noureddine Boudriga. 2010. Optimized methods for inserting and deleting records and data retrieving in quantum database. In *2010 12th International Conference on Transparent Optical Networks*. IEEE, 1–5.

[13] Phillip Kaye, Raymond Laflamme, and Michele Mosca. 2006. *An introduction to quantum computing*. OUP Oxford.

[14] Nitin Nayak, Jan Rehfeld, Tobias Winker, Benjamin Warnke, Umut Çalikyilmaz, and Sven Groppe. 2023. Constructing Optimal Bushy Join Trees by Solving QUBO Problems on Quantum Hardware and Simulators. In *Proceedings of the International Workshop on Big Data in Emergent Distributed Environments*. 1–7.

[15] Michael A Nielsen and Isaac L Chuang. 2010. *Quantum computation and quantum information*. Cambridge university press.

[16] Maryann Njorbuenwu, Bobby Swar, and Pavol Zavarsky. 2019. A survey on the impacts of quantum computers on information security. In *2019 2nd International conference on data intelligence and security (ICDIS)*. IEEE, 212–218.

[17] Yuri Ozhigov. 1997. Protection of information in quantum qatabases. *arXiv preprint quant-ph/9712016* (1997).

[18] Chao-Yang Pang, Ri-Gui Zhou, Cong-Bao Ding, and Ben-Qiong Hu. 2013. Quantum search algorithm for set operation. *Quantum information processing* 12, 1 (2013), 481–492.

[19] John H Reif. 2009. Quantum Information Processing: Algorithms, Technologies and Challenges. *BIO-INSPIRED COMPUTING,(EDITED BY MM ESHAGHIAN-WILNER* (2009).

[20] Sudip Roy, Lucja Kot, and Christoph Koch. 2013. Quantum databases. In *Proc. CIDR*.

[21] Manuel Schönberger, Stefanie Scherzinger, and Wolfgang Mauerer. 2023. Ready to leap (by co-design)? join order optimisation on quantum hardware. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–27.

[22] Manuel Schönberger, Immanuel Trummer, and Wolfgang Mauerer. 2023. Quantum Optimisation of General Join Trees. In *Joint Workshops at 49th International Conference on Very Large Data Bases (VLDBW'23)—International Workshop on Quantum Data Science and Management (QDSM'23)*.

[23] Mike Stonebraker, Daniel J Abadi, Adam Batkin, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amerson Lin, Sam Madden, Elizabeth O'Neil, et al. 2018. C-store: a column-oriented DBMS. In *Making Databases Work: the Pragmatic Wisdom of Michael Stonebraker*. 491–518.

[24] Immanuel Trummer and Christoph Koch. 2015. Multiple query optimization on the D-Wave 2X adiabatic quantum computer. *arXiv preprint arXiv:1510.06437* (2015).

[25] Manuela Weigold, Johanna Barzen, Frank Leymann, and Marie Salm. 2021. Encoding patterns for quantum algorithms. *IET Quantum Communication* 2, 4 (2021), 141–152.

[26] Tobias Winker, Sven Groppe, Valter Uotila, Zhengtong Yan, Jiaheng Lu, Maja Franz, and Wolfgang Mauerer. 2023. Quantum Machine Learning: Foundation, New Techniques, and Opportunities for Database Research. In *Companion of the 2023 International Conference on Management of Data*. 45–52.

[27] Ahmed Younes. 2007. Database manipulation on quantum computers. *arXiv preprint arXiv:0705.4303* (2007).

[28] Ahmed Younes and Julian Miller. 2003. Automated method for building CNOT based quantum circuits for Boolean functions. *arXiv preprint quant-ph/0304099* (2003).

[29] Gongsheng Yuan. 2020. How the quantum-inspired framework supports keyword searches on multi-model databases. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 3257–3260.

[30] Gongsheng Yuan, Yuxing Chen, Jiaheng Lu, Sai Wu, Zhiwei Ye, Ling Qian, and Gang Chen. 2024. Quantum Computing for Databases: Overview and Challenges. arXiv:2405.12511 [cs.DB] https://arxiv.org/abs/2405.12511

[31] Gongsheng Yuan, Jiaheng Lu, Yuxing Chen, Sai Wu, Chang Yao, Zhengtong Yan, Tuodu Li, and Gang Chen. 2023. Quantum Computing for Databases: A Short Survey and Vision. (2023).

[32] Gongsheng Yuan, Jiaheng Lu, and Peifeng Su. 2021. Quantum-Inspired Keyword Search on Multi-model Databases. In *Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11–14, 2021, Proceedings, Part II 26*. Springer, 585–602.

A EXPERIMENT RESULT

Table 5 shows more results. **Integrity Ratio** means that the records of relational tables included in the quantum circuit measurement

Table 5: Results of simulators and real universal quantum computers

Storage Structure	Schema	dataQuantity	Backend	Noise	Qubits	Depth	Gates (MCT)	Valid Ratio	Integrity Ratio
QROS	(2×2)	4	qasm_simulator	No	4	5	7(3)	1.00	1.00
	(3×1,3×2)	9	qasm_simulator	No	14	30	53(36)	1.00	1.00
	(2×1,3×1,2×3)	11	qasm_simulator	No	15	16	36(18)	1.00	1.00
	(2×2)	4	aer_simulator	Yes	4	5	7(3)	0.94	1.00
	(3×1,3×2)	9	aer_simulator	Yes	14	30	53(36)	0.82	1.00
	(2×1,3×1,2×3)	11	aer_simulator	Yes	15	16	36(18)	0.78	1.00
	(2×2)	4	ibm_brisbane	Yes	127	14	7(3)	0.96	1.00
	(3×1,3×2)	9	ibm_brisbane	Yes	127	534	53(36)	0.00	0.00
	(2×2,3×2,3×4)	22	ibm_brisbane	Yes	127	654	68(42)	0.00	0.00
	(2×1,1×2)	4	ibm_brisbane	Yes	127	6	8(2)	0.00	1.00
	(2×1,1×2)	4	ibm_lagos	Yes	7	5	8(2)	0.00	0.00
	(2×1,1×2)	4	ibm_nairobi	Yes	7	5	8(2)	0.001	0.50
	(2×1,1×2)	4	ibm_perth	Yes	7	5	8(2)	0.00025	0.50
	(2×2)	4	ibm_lagos	Yes	7	7	7(3)	0.96	1.00
	(2×2)	4	ibm_nairobi	Yes	7	7	7(3)	0.91	1.00
	(2×2)	4	ibm_perth	Yes	7	7	7(3)	0.91	1.00
QCOS	(2×2)	4	qasm_simulator	No	5	13	14(8)	1.00	1.00
	(3×1,3×2)	9	qasm_simulator	No	8	160	50(38)	1.00	1.00
	(2×1,3×1,2×3)	11	qasm_simulator	No	9	357	146(104)	1.00	1.00
	(2×2)	4	aer_simulator	Yes	5	13	14(8)	0.89	1.00
	(3×1,3×2)	9	aer_simulator	Yes	8	160	50(38)	0.90	1.00
	(2×1,3×1,2×3)	11	aer_simulator	Yes	9	357	146(104)	0.87	1.00
	(2×2)	4	ibm_brisbane	Yes	127	370	14(8)	0.49	1.00
	(3×1,3×2)	9	ibm_brisbane	Yes	127	6814	52(40)	0.07	1.00
	(2×2,3×2,3×4)	22	ibm_brisbane	Yes	127	85164	202(144)	0.08	1.00
	(2×1,1×2)	4	ibm_brisbane	Yes	127	390	15(6)	0.35	1.00
	(2×1,1×2)	4	ibm_lagos	Yes	7	153	15(6)	0.34	1.00
	(2×1,1×2)	4	ibm_nairobi	Yes	7	151	15(6)	0.30	1.00
	(2×1,1×2)	4	ibm_perth	Yes	7	153	15(6)	0.29	1.00
	(2×2)	4	ibm_lagos	Yes	7	173	14(8)	0.67	1.00
	(2×2)	4	ibm_nairobi	Yes	7	173	14(8)	0.56	1.00
	(2×2)	4	ibm_perth	Yes	7	173	14(8)	0.50	1.00

results account for the proportion of the total number of relationship table records. To be specific, assume that the measurement results of the quantum circuit have 190 records, 95 of which are records in the original relational table. The original relational table has 100 records, then **Valid Ratio** is 50% and **Integrity Ratio** is 95%.

A.1 Simulator Result

Table 5 presents the outcomes on the simulators provided by IBM Qiskit. As the schema complexity and **Data Quantity** increase, QCOS consumes fewer qubits compared to QROS. However, the cost associated with quantum gates and *MCT* gates for QCOS significantly surpasses that of QROS. Moreover, the circuit depth of

QCOS substantially exceeds that of QROS, implying that the implementation of QCOS may employ a more intricate circuit topology. It calls for further optimization. From the perspective of the **Valid Ratio** (i.e., a ratio of correct data to the total data in the measurement results), QCOS is always better than QROS in the presence of noise, especially as **Data Quantity** increases. This phenomenon may be attributed to the higher qubit cost of QROS, which increases the probability of noisy basic states. Finally, the **Integrity Ratio** (i.e., the proportion of measurement results belonging to the dataset out of the dataset) of both QCOS and QROS is 1, indicating that their measurement results contain all information about the datasets.

A.2 Real Device Result

As shown in Table 5, the circuit depth and the number of quantum gates on real devices are much higher than those on simulators for the same datasets. This difference may be caused by the physical topology of the device. The results show that the dataset with a smaller **Data Quantity** has a higher accuracy, while larger and more complex data sets are more susceptible to noise. Besides, the accuracy of QROS is much higher than that of QCOS for the dataset with a smaller **Data Quantity**. As for larger datasets, although QCOS has some valid data and QROS has none, the **Valid Ratio** for QCOS is very low.

We conducted experiments on three 7-qubit and one 127-qubit IBM real machines with two smaller datasets. The data in Table 5 shows that machines with the same number of qubits exhibit the same circuit depth and gate cost. Performance metrics such as the **Valid Ratio** demonstrate similar trends. Although a 127-qubit machine could handle greater depth, its performance is still the same as that of a 7-qubit machine. When the same **Data Quantity** is stored in two separate tables instead of a single table, employing QCOS and QROS to process that data demonstrates significantly poorer performance, especially for QROS. Finally, the gap between the outcomes of aer_simulator and ibm_brisbane reveals the limitations of current-stage quantum hardware.