Qute：迈向量子原生数据库

陈牧之 上海交通大学 inefable@sjtu.edu.cn
Xuanhe Zhou* 上海交通大学 zhouxuanhe@sjtu.edu.cn
周伟 上海交通大学 wzdb@sjtu.edu.cn
徐邦瑞 唐苏睿 上海交通大学 上海交通大学 dreameter@sjtu.edu.cn tangsurui@sjtu.edu.cn
李国良 清华大学 liguoliang@tsinghua.edu.cn
何秉胜 新加坡国立大学 dcsheb@nus.edu.sg
Yeye He 微软公司 yeyehe@microsoft.com
Yitong Song 香港浸会大学 ytsong@hkbu.edu.hk
Fan Wu 上海交通大学 fwu@cs.sjtu.edu.cn

arXiv:2602.14699v1 [cs.DB] 16 Feb 2026

摘要

本文展望了一种量子数据库（Qute），将量子计算视为一等执行选项。不同于以往基于模拟的方法——这些方法要么在经典计算机上运行量子算法，要么改造现有数据库以进行量子模拟，Qute 则（i）将扩展形式的 SQL 编译为门效率高的量子电路；（ii）采用混合优化器，动态选择量子或经典执行计划；（iii）引入选择性量子索引；（iv）设计保真度保持存储，以缓解当前的量子比特限制。我们还提出了迈向量子原生数据库的三阶段演进路线图。最后，通过将Qute 部署在一台真实的量子处理器（源自_wukong）上，我们展示了它在大规模场景下优于经典基线，并在以下地址发布了开源原型：*https://github.com/weAIDB/Qute*。

1 引言

量子数据库旨在利用量子计算机加速那些对经典计算机而言日益难以处理的工作负载。例如，列式 OLAP 系统中的分析型查询处理通常依赖于对无索引表进行线性扫描 [46]。通过利用纠缠量子比特叠加态和格罗弗搜索等量子算法 [45], , 量子数据库能够显著加速此类查询。近期研究主要沿着两个方向展开。首先，面向数据库的量子计算 将量子算法应用于经典数据库，以执行查询优化 [26, 38], 、索引调优 [17], 和近似查询处理 [43]等任务。然而，这些方法仍基于模拟，缺乏全面的量子集成。其次，面向量子计算的数据库 利用经典数据库技术来管理量子工作负载，

例如以表格形式编码量子电路，但未考虑数据库端的优化 [18, 23]。

为了解决这些局限，我们致力于构建一种端到端的量子原生数据库，将量子计算视为贯穿查询解析、规划和执行各个环节的一流执行底座，而非孤立的加速器。然而，这面临四项主要挑战。首先，必须重新构建 SQL 算子的表示方式，使其能够自然映射到量子原语（例如叠加态、干涉）。其次，优化器必须针对易受噪声影响的量子算子进行规划；这些算子的延迟、成功概率和近似误差与经典优化器中的情况存在根本差异。第三，量子比特预算有限，因此必须有选择地探测数据集中的一小部分

*Xuanhe Zhou 是通讯作者。

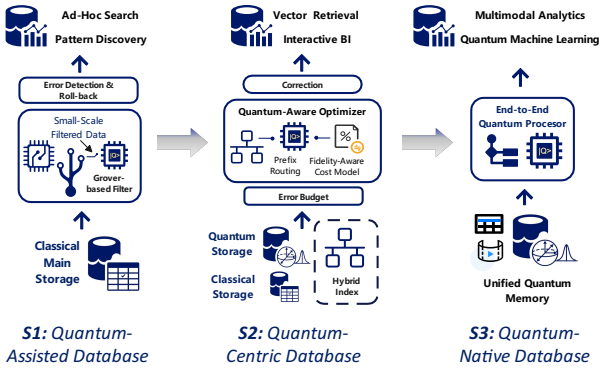


图1：量子数据库的潜在演进。

以进行量子执行。最后，必须以能够保持保真度的格式存储量子活跃属性，以维持较高的可用性。

本文介绍了 Qute，这是一种集成式量子数据库架构，旨在应对上述挑战。首先，Qute 通过三层中间表示，将扩展 SQL 查询编译为量子门高效电路。其次，它采用结合量子算子与经典算子的混合优化器。第三，它将高开销算子（例如相似性连接）卸载到量子电路上，并将计算编码为概率振幅。第四，为解决量子比特数等规模限制，Qute 引入了基于增强型混合树结构的选择性量子索引。第五，它采用保真度感知数据格式和存储引擎，支持压缩张量网络。

如图1所示，我们设想量子原生数据库将经历三个发展阶段。第一阶段，量子计算作为经典系统的协处理器，数据仍存储在经典内存中，仅将小部分数据加载到量子寄存器，以执行数据过滤等基本任务。此阶段缺乏索引加速，并依赖回滚机制来确保在噪声环境下的正确性。第二阶段，更多数据可以存储在量子内存中，从而支持经典–量子混合索引，以及连接和聚合等量子算子。第三阶段，随着可扩展容错硬件的发展，所有数据（包括视频等多模态格式）都可以存储在量子内存中，从而实现完全量子原生的存储、索引和处理。

2 Preliminary

我们介绍支撑 Qute 的量子概念，包括：(1) 量子比特，(2) 量子门，以及 (3) 量子电路。

量子比特。经典计算机将信息编码为 {0，1} 中的确定性比特，并通过逐一显式枚举和评估候选解来解决计算任务。相比之下，量子计算机使用量子比特编码信息，其状态不受单一确定性值的限制，而是可以处于经典基态 $|0\rangle$ 和 $|1\rangle$ 的叠加态 中。经过测量后，量子比特会坍缩为确定值 0 或 1 [32]。形式上，量子比特的状态为 $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$，其中振幅 α 和 β 编码了大小和相位；测量得到 0 或 1 的概率分别为 $|\alpha|^2$ 和 $|\beta|^2$，且满足 $|\alpha|^2 + |\beta|^2 = 1$。借助量子比特，可以隐式表示大量候选解 [2]，并利用适当的量子门和量子电路放大目标解出现的概率，从而降低搜索复杂度 [15]。

量子门。量子门是一种以受控方式改变量子比特状态的操作，能够增强正确的可能性，同时抑制错误的可能性。量子门通常分为两类：单量子比特门 和多量子比特门。• ***(1)*** 单量子比特门通过操控单个量子比特的振幅（例如 X 轴和 Y 轴旋转及翻转）与相位（例如 Z 轴旋转），实现数据表示和量子计算所必需的操作。对于状态制备，(*i*)*Hadamard*（*H*）门将基态映射为 $|0\rangle$ 和 $|1\rangle$ 的等幅叠加，为量子计算提供~~均匀的初始表示；~~ (*ii*) ***X*** 轴和 ***Y*** 轴旋转（即 $R_X(\theta)$ 和 $R_Y(\theta)$）通过调整量子比特在 $|0\rangle$ 与 $|1\rangle$ 之间的振幅贡献来编码行值；(*iii*)*Pauli-X* 翻转使量子比特在 $|0\rangle$ 与 $|1\rangle$ 之间切换，用于标记特定的基态。对于相位调整，*Z* 轴旋转门（即 $R_Z(\theta)$）改变 $|0\rangle$ 与 $|1\rangle$ 之间的相对相位，这决定了不同可能性是相互增强还是相互抵消（从而提高目标行的概率）。~~对于数值编码，参数化旋转~~ $R_\alpha(\theta)$，其中 $\alpha \in \{X, Y, Z\}$，可同时调整幅度和相位，从而将数值信息嵌入 [30, 32]。

• ***(2)*** 多量子比特门执行条件操作，并在量子比特之间建立纠缠，从而形成它们之间的关联。这种关联对于量子并行以及计算加速至关重要，而这些是经典计算机无法实现的。多量子比特门大致可分为三类：(*i*) 受控门仅在控制量子比特处于特定状态时作用于目标量子比特；(*ii*) 纠缠门将多个量子比特关联起来，以产生非经典关联；(*iii*) 置换门交换量子比特的状态，以重新排列数据。例如，*Controlled-SWAP (CSWAP)* 门结合了受控操作和置换操作，仅当控制量子比特处于特定状态时，才交换两个目标量子比特的状态 [29]。

量子电路。量子电路通过组织一系列量子门来实现量子算法并生成所需结果。我们使用一个具有代表性的量子电路（即 *Grover* 算法 [16]）来说明其加速数据过滤的潜力。给定数据集（大小为 $N = |\mathrm{X}|$）上的布尔谓词 $f: \mathrm{X} \rightarrow \{0,1\}$（位于 SQL WHERE子句中），数据过滤会找出所有满足 $f(x) = 1$ 的记录 $x \in \mathrm{X}$。该电路首先通过 H 门对整个数据集进行均匀编码，然后组织一系列包含以下两个算子的量子门。

• ***(1)*** 预言机 ***(O_f)***：预言机充当谓词求值器，通过量子算子实现，用于编码查询条件

Table 1: Classical vs. Quantum Database Components.

Component	Classical Database	Quantum Database
Compiler	Generate efficient machine code for Classic 执行（中央处理器、图形处理器等）。	生成可直接执行的电路 于量子后端上。
算子	执行逐元组处理或向量- 化处理（中央处理器开销 ↓）。	Replace tuple scans with amplitude and interference operations.
Plan Optimizer	Select the lowest-cost deterministic execution path.	Dynamically balance quantum speedup vs. noise / success rate.
Data Indexing	Accelerate lookup operations.	Shrink the candidate set that must be loaded into quantum computer.
Data Format	字节可寻址且易于解析。	将数据编码为量子比特状态。
存储引擎	确保持久性和高效的输入/输出。	维护可重构的大规模 数据，并采用压缩技术。

（例如，$f(x) = 1$）。作用于叠加态 $|x\rangle$ 时，它通过相位反转标记满足谓词的记录，即 $O_f|x\rangle = (-1)^{f(x)}|x\rangle$。

• ***(2)*** 扩散算子（***D***）：该算子通过将概率振幅关于其均值进行反射来提高成功概率，增大标记状态的振幅并抑制其他状态的振幅。其实现方式是对所有量子比特应用 H 门，对 $|0\cdots0\rangle$ 状态执行多量子比特相位反转，然后反向执行哈达玛变换，即 $D = 2|s\rangle\langle s| - I$。

迭代应用算子 $G = DO_f$ 共 $k \approx \frac{\pi}{4}\sqrt{N/M}$ 步（其中 M 为匹配记录数）后，测量以较高概率得到匹配记录。不同于需要进行 $O(N)$ 次谓词求值的经典线性扫描，该电路仅需进行 $O(\sqrt{N})$ 次求值（即最坏情况下为 $M = 1$）即可找到匹配记录，凸显了其在对未索引数据进行滤波方面的优势。

3 系统概述

我们介绍经典数据库与量子数据库之间的差异（表 1），并概述 Qute 的关键组件（图 2）。

设计原则。量子数据库遵循若干不同于经典数据库的设计原则。(1) 尽可能缩小量子活跃足迹，以最大限度减少量子数据加载（例如，通过选择性索引、预过滤和按需编码）。(2) 利用振幅级并行，而非逐元组处理（例如，将其重新表述为量子算子）。(3) 采用保真度感知优化策略（例如，考虑量子算子中可能存在的噪声）。(4) 保持混合可执行性，使量子路径和经典路径都能正常运行，从而应对硬件噪声和资源限制。

量子 SQL 编译器。Qute 集成了量子原生编译栈：(1) 我们既处理标准关系算子（例如滤波、聚合），也支持采样估计和概率评估等适用于量子的函数，这些函数可通过振幅估计等原语实现 [36]。请注意，我们可以通过模块化电路生成器 [33]为每个算子自动生成门高效电路；(2) 我们采用三级 LLVM 风格的中间表示（逻辑中间表示、量子扩展中间表示和物理电路中间表示），支持将声明式 SQL 逐步降低为可执行的量子电路 [37]；(3) 我们维护一个动态重写规则目录，借助叠加态启用的概率分支和基于干涉的相关性检测等量子特性，为复杂连接和多谓词过滤查询发掘有用的转换。

量子计划优化器。Qute 使用紧凑概况（qT_q，P_q，ε_q）来描述每个量子算子，总结其延迟、

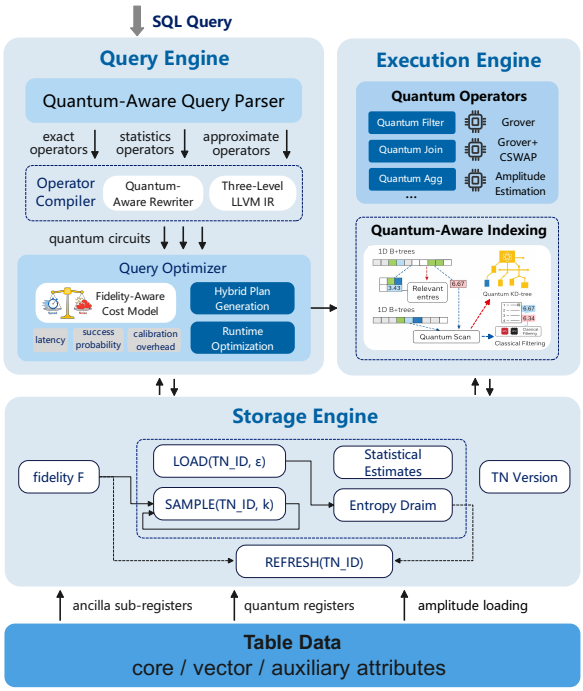


Figure 2: Example Quantum Database System Design.

成功概率和近似误差，使优化器能够将其与经典算子直接比较。借助这些概况，执行计划会为符合条件的算子同时嵌入量子和经典实现，从而使域选择保持灵活。运行时，Qute 会监控质量和稳定性，并通过增加采样次数、切换到更稳健的电路变体，或在量子性能下降时回退到经典路径来适应噪声。

量子可加速算子。 Qute 将代价高昂的操作卸载到量子子程序中，这些子程序将计算编码为概率振幅，而非逐行执行 [10, 41]。例如，对于相似性连接，Qute 通过将向量编码为量子态并利用基于干涉的重叠估算其内积，从而绕过按维度遍历。

量子感知索引。 为解决量子内存瓶颈，我们开发了支持选择性量子探测的索引结构，即在分配量子资源之前，仅探测与查询相关的索引条目。我们首先对一维 B^+-树进行轻量级量子扫描，以快速缩小候选集。当筛选后只剩少量候选项时，剩余工作便通过经典过滤完成。然而，如果仍有大量候选项，我们就转而使用量子 KD 树对多个维度进行联合索引，从而无需穷举扫描所有行即可进行更积极的剪枝。

量子数据格式。 *(1)* 基编码：核心行标识符（如 ID_{row}）会映射到计算基态，从而实现均匀叠加态和并行预言机处理。*(2)* 振幅编码：数据向量（如嵌入）通过旋转门编码为状态振幅，将数值压缩为与行索引对应的概率分布。*(3)* 控制编码：元数据和标志存储在充当控制线的辅助量子比特中，从而可根据行级属性执行条件门。

量子存储引擎。 与基于页的设计不同，[24], Qute 采用以状态为中心的方法。量子活跃属性以压缩张量网络（TN）的形式存储，充当逻辑页，支持在保真度约束下进行有界重建和采样。该引擎提供保真度感知原语，以实现受控近似，同时跟踪熵和噪声：LOAD(TN_ID, ϵ) 用于近似访问，SAMPLE(TN_ID, k) 用于有界采样，而 REFRESH(TN_ID) 则在保真度下降超过阈值时恢复保真度。经典元数据单独存储，以确保可恢复性。

4 量子可加速算子

基于第 2 节介绍的基础电路，我们设计了针对数据库核心操作的量子电路。每个算子都经过重新设计，以利用量子并行，同时保留关系语义。表 2 总结了量子可加速算子，下面我们将介绍其中三种代表性类型。

4.1 量子数据过滤

大规模数据过滤是数据库的一项基本操作，但在经典系统上处理稀疏且未建立索引的数据时，由于需要进行穷举式全表扫描，会产生较高的延迟。为克服这一瓶颈，Qute 将格罗弗算法改造为量子数据过滤算子。我们设计了一种定制量子电路，通过实现一系列逻辑门和扩散门，以加速的量子搜索取代经典线性扫描。

电路设计。 具体而言，我们将行标识符（ID_{row}）域映射为搜索空间，并将 ϕ 编译为量子预言机 O_ϕ，用于标记满足条件的行（如第 2 节中的 O_f）。复合 SQL 谓词通过辅助比特辅助编码转换为相位标记逻辑。例如，像 `age > 30 AND city = 'NY'` 这样的谓词，会先将各个子条件计算到辅助量子比特中，再使用多重受控操作将它们组合起来，从而标记最终目标状态。Grover 迭代放大匹配项 ID_{row} 的振幅后，Qute 会执行经典协调，即对匹配的 ID_{row} 去重，并根据 ϕ 重新检查这些项，以确保正确性。

4.2 量子相似性连接

相似性连接是数据库和多模态数据检索中的基础操作 [3, 8, 39, 44]。其目标是从两个数据集中检索出所有距离在给定阈值范围内的向量对。相似性连接的核心操作是距离计算（例如内积），用于确定与每个输入向量相似（即距离较小）的向量集合。经典计算机通常需要耗费大量资源来计算向量之间的内积（例如，由于需要对大量向量对反复进行逐维乘法和距离评估）。相比之下，Qute 将向量编码为量子态，并在不显式遍历所有维度或候选向量对的情况下计算内积。

电路设计。 如图 3 所示，给定两个归一化向量 $\vec{x}, \vec{y} \in \mathbb{R}^d$，Qute 首先将它们编码为量子态 $|x\rangle$ 和 $|y\rangle$。每个向量都按照第 3 节所述的向量到量子态映射，通过参数化单量子比特旋转（例如 $R_Y(\theta)$）转换为量子态。这种编码确保了 $\vec{x}$ 与 $\vec{y}$ 之间的内积会反映在

表 2：可由量子加速的关系算子（M 和 N 分别为总行数和符合条件的行数； d 为向量维度；ϵ 为精度； n 和 q_* 分别为数据寄存器和精度寄存器的量子比特数； a_* 为辅助比特； b 为带宽； D 为门深度）。

Type	算子 (SQL)	典型场景	候选量子算法	Complexity		Cost Model	
				Quantum	Classical (without index)	Qubit Num	Gate Depth (per-circuit call)
Filter	Equality Filter	`WHERE column = value`	Grover [16] (Search)	$O(\sqrt{N})$	$O(N)$	$n + a_{\text{orc}}$	$O\Big((D_{\text{orc}} + D_{\text{diff}}) \cdot \sqrt{N/M}\Big)$
	范围过滤器	WHERE 列 > 值	格罗弗 [16] (阈值预言机)	$O(\sqrt{N})$	$O(N)$	$n + b + a_{\text{cmp}}$	$O\Big((D_{\text{cmp}}(b) + D_{\text{diff}}) \cdot \sqrt{N/M}\Big)$
	LIKE	WHERE 列 LIKE 'val%'	格罗弗 [16] (前缀匹配预言机)	$O(\sqrt{N})$	$O(N)$	$n + b + a_{\text{pref}}$	$O\Big((D_{\text{pref}}(b) + D_{\text{diff}}) \cdot \sqrt{N/M}\Big)$
	EXISTS	存在性检查	格罗弗 [16] （量子计数）	$O(\sqrt{N})$	$O(N)$	$n + a_{\text{orc}}$	$O\Big((D_{\text{orc}} + D_{\text{diff}}) \cdot \sqrt{N}\Big)$
Join	等值连接	列1 = 列2	格罗弗 [16] (索引探测)	$O(\sqrt{N_2})$ / 探测	$O(N_1 \cdot N_2)$	$n_2 + b_{\text{key}} + a_{\text{orc}}$	$O\big((D_{\text{idx}} + D_{\text{key}} + D_{\text{diff}}(n_2)) \cdot \sqrt{N_2}\big)$ /探测
	非等值连接	列1 ≤列2	格罗弗 [16] (比较预言机)	$O(\sqrt{N_2})$ / 探测	$O(N_1 \cdot N_2)$	$n_2 + b_{\text{key}} + b + a_{\text{cmp}}$	$O\big((D_{\text{cmp}} + D_{\text{diff}}(n_2)) \cdot \sqrt{N_2}\big)$ /探测
	相似性连接	相似性搜索	格罗弗 [16] (SWAP 测试)	$O(\sqrt{N})$	$O(d \cdot N)$	$2q + a_{\text{prep}} + 1$	$O(2 \cdot D_{\text{prep}}(d) + O(q))$
聚合	MIN	全局最小值搜索	Dürr–Høyer 最小值查找 g [9]	$O(\sqrt{N})$	$O(N)$	$n + b + a_{\text{cmp}}$	$O\Big((D_{\text{cmp}} + D_{\text{diff}}) \cdot \sqrt{N}\Big)$
	计数	基数估计	振幅估计 [6]	$O(1/\epsilon)$	$O(N)$	$n + a_{\text{orc}} + q_\epsilon$	$O((D_{\text{orc}} + D_{\text{diff}}) \cdot (1/\epsilon))$
	SUM	总值计算	归一化 + 振幅估计 [6]	$O(1/\epsilon)$	$O(N)$	$n + b_v + a_{\text{rot}} + q_\epsilon$	$O((D_{\text{load}} + D_{\text{rot}} + D_{\text{diff}}) \cdot (1/\epsilon))$
	AVG	平均值计算	归一化 + 振幅估计 [6]	$O(1/\epsilon)$	$O(N)$	$n + b_v + a_{\text{rot}} + q_\epsilon$	$O((D_{\text{load}} + D_{\text{rot}} + D_{\text{diff}}) \cdot (1/\epsilon))$
其他	采样	近似查询	振幅估计 [6]	$O(\sqrt{N})$	$O(N)$	$n + a_{\text{orc}}$	$O\Big((D_{\text{orc}} + D_{\text{diff}}) \cdot \sqrt{N/M}\Big)$

对应量子态的重叠度中。为估计这种重叠度，Qute 使用 *SWAP* 测试 [10], ，这是一种用于内积估计的标准量子子程序。具体而言，首先通过 H 门将辅助量子比特初始化为均匀叠加态，随后在辅助量子比特的控制下，对两个数据寄存器执行 CSWAP 操作。然后再对辅助量子比特施加第二个 H 门。测量辅助量子比特后，观测到结果 $|0\rangle$ 的概率为 $\Pr[\text{辅助比特} = 0] = (1+|\langle x|y\rangle|^2)/2$。通过反复执行电路并测量辅助量子比特，Qute 根据实际测量频率估计 $|\langle x|y\rangle|^2$。经典后处理会将测量结果转换为近似内积，以识别向量对。

4.3 量子聚合

经典聚合（例如 SUM）通过遍历所有符合条件的行并累加其值来计算结果。当符合条件的数据集较大时，由于计算量随行数线性增长，因此延迟较高。量子系统将信息编码为概率振幅，因此无法直接复现经典累加器。为此，Qute 将聚合重新表述为概率估计任务，使输入/输出不会在采用相干向量访问（例如通过量子随机存取存储器）的情况下主导端到端成本。通过将数值映射到目标量子比特的观测概率，Qute 利用振幅估计 [6] 高效计算总和，无需逐行遍历。

电路设计。 Qute 首先对所有 N 候选行标识符制备均匀叠加态。对于每个行标识符 $|x\rangle$，访问对应的值 $v(x)$，并用它来控制一个专用的标志量子比特，即*Good* 量子比特。具体而言，施加受控旋转，使测量到 Good 量子比特为 $|1\rangle$ 的概率与归一化后的值 $v(x)/V_{\max}$ 成正比，

其中 $V_{\max}$ 是聚合值的已知上界。直观而言，值较大的行对观测到 Good 标志的概率贡献更大。测得 Good 量子比特为 $|1\rangle$ 的总体概率等于所有行的归一化值的平均值，即 $Pr[\text{Good} = 1] = \frac{1}{N}\sum_{x=0}^{N-1} v(x)/V_{\text{最大值}}$。通过使用振幅估计对该值进行估算，并乘以 N 和 $V_{\max}$ 进行缩放，Qute 可得出所需结果：$\text{SUM}(v) = N \cdot V_{\max} \cdot \Pr[\text{Good} = 1]$。

5 量子感知索引结构

受量子比特容量和电路深度的限制，现有量子计算机只能将一小部分数据编码到叠加态中 [18], ，因此选择性加载数据至关重要。现有方法仅将相关的 B⁺ 树叶节点加载到量子处理器中，但这种策略难以推广到多维查询，且性能会显著下降。因此，我们提出一种量子加速索引策略，通过轻量级探测提前缩小查询范围，同时支持一维和多维工作负载。

我们的策略将选定维度上的量子探测与经典后过滤相结合。对于多列谓词（图 4），我们首先探测基于选定维度构建的若干一维量子 B⁺-树。此探测会找出完全包含于或部分重叠于查询范围 [27] 的节点。例如，对于查询 q_1： $d_1 \in [5, 8]$，量子探测会返回满足维度 d_1 上谓词的 $k_1 = 4$ 个元组。在所有被探测的 B⁺-树返回的结果集中，我们选取最小的一个作为候选集，并将其大小记为 k_s。根据 k_s，我们动态地在两种策略之间进行选择。

- **(1) 经典后过滤。** 如果候选数量较少（即 $k_s = O(\log N)$），我们会对剩余的 $d - 1$ 个维度进行经典验证。对于查询 q_1，我们使用次级谓词 $d_2 \in [1, 6]$ 进一步过滤检索到的 4 行。其开销为 $O(k_s \cdot (d - 1))$；由于候选集大小较小，因此效率较高。

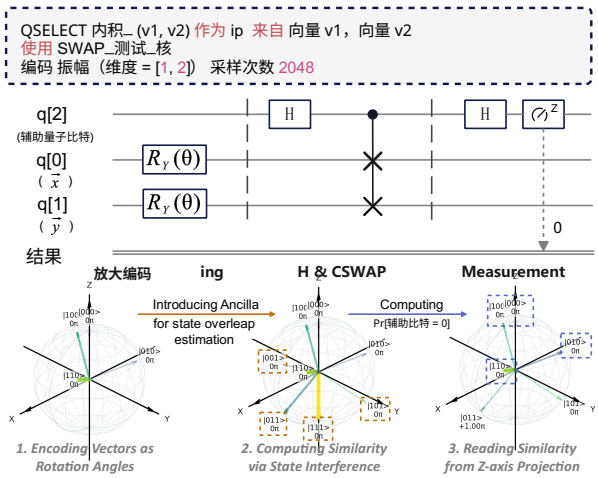


图 3：基于 CSWAP 门的内积计算。

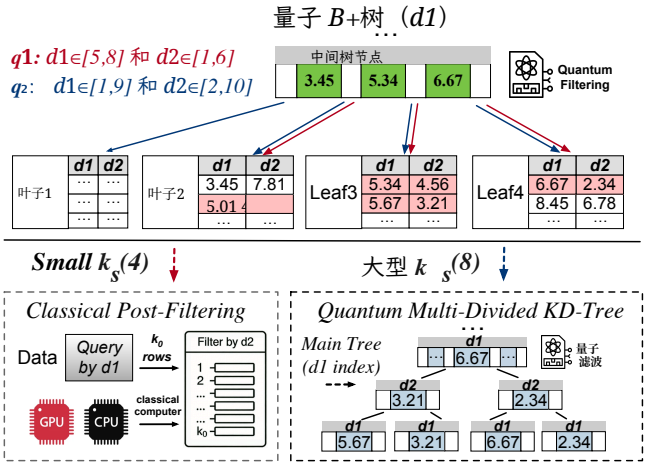


图 4：量子感知多维索引。

• **(2) 量子多划分 KD 树搜索。** 如果 k_s 较大，经典过滤的效果就会下降。此时我们会采用一种量子多划分 KD 树，对多个维度进行联合索引。该结构通过沿不同维度递归划分数据，并在每一层应用量子辅助探测，在验证前显著缩小候选集，因此更适用于大规模 k_s 场景。

此外，对于涉及析取条件的查询（例如，$d_1 \in [5,8] \vee d_2 \in [2,5]$），我们会独立探测每个维度，并合并得到的叠加态。

6 量子感知优化器

经典查询优化器依赖确定性假设，无法捕捉量子硬件固有的随机性和噪声。Qute 通过将量子算子视为具有精度界限的概率性基本操作，解决了这一局限，从而将优化目标从单纯最小化延迟转向速度与可靠性之间的稳健权衡。

估计模型。 该成本模型扩展了经典优化：它将量子算子视为具有精度界限的随机性基本操作，而非确定性函数。优化器使用一个紧凑的元组（T_q，P_q，ϵ_q）评估每个算子，该元组涵盖其延迟、成功概率和近似误差；这些指标直接依据电路结构和硬件特性进行量化。每个因素都被明确纳入目标函数，使优化器能够根据真实硬件行为，在执行速度与预期可靠性之间进行权衡。

• **（1）时间估计（T_q）。** 执行时间根据拓扑感知路由后的电路分层调度得出。对于各层$L = \{L_1, \ldots\ldots, L_K\}$，$T_q = \sum_{k=1}^{K} \left(\max_{g \in L_k} t_g + t_{\text{ctrl}} \right)$，其中 t_g 是每个门经过校准后的持续时间，t_{ctrl} 表示层间的控制与同步开销。执行时间受两个延迟因素影响：(i) 有限的量子比特连通性所导致的串行化，以及 (ii) SWAP 引起的深度膨胀；两者均用于评估量子算子是否优于经典算子。

• **（2）噪声与成功概率（P_q）。** 量子成功概率通过电路层级的门错误与退相干组合来估算。对于每一层，$p_k = \left(1 - \sum_{g \in L_k} \epsilon_g\right) \cdot \exp\left(-\frac{t_k}{T_{2,\text{eff}}}\right)$，其中$\epsilon_g$ 硬件 —报告的门错误率和 T_2^{eff} 一个有效的相干性界。算子级成功概率近似为 $P_q = \prod_{k=1}^{K} p_k$。它将计划质量与电路深度和布局直接关联起来：电路越深或路由越不理想，成功概率就会受到乘法式衰减，从而引导优化器剪除过于脆弱的计划。

• **(3) 混合执行误差 (ϵ_q)。** 每个量子算子都配有一个确定性的经典回退。优化器会将其预期运行时间评估为 $\mathbb{E}[T_q] = P_q \cdot T_{\text{量子}} + (1 - P_q) \cdot T_{\text{经典}}$，确保错误或失败的执行不会影响结果的正确性。近似误差ϵ_q根据算子的振幅、估计精度或采样预算得出，并根据查询精度要求作为约束或惩罚纳入考量。优化器利用运行时观测结果更新参数，从而能够自适应地优化计划选择。

通过将（T_q、P_q、ϵ_q）的每个组件都建立在明确的电路和硬件特征之上，该模型为量子-经典计划成本建模提供了具体且可执行的基础。

混合计划生成。 在 Qute 中，每个计划都同时编码所选算子的量子和经典实现，在保持语义等价的同时，提供明确的替代方案，以权衡成本与质量。重要的是，跨域协调成本（例如经典运行时与量子运行时之间的数据移动）被明确纳入优化器的目标函数，从而避免成本估计过于乐观。此外，Qute 支持对执行域选择进行延迟绑定，使运行时调度器能够根据实时条件（例如后端校准状态、队列延迟）选择最合适的实现。

自适应运行时优化。 为确保在存在量子噪声和随机故障的情况下仍能可靠执行，Qute 将自适应运行时优化作为一项一等机制。当量子执行路径未能达到声明的质量或延迟阈值时，系统会动态分配资源，例如增加采样次数、切换到更稳健的电路变体，或回退到经典计算。重要的是，所有此类调整都遵循有界误差语义：回退路径会预先验证其正确性，且所有运行时决策均受优化器声明的误差和延迟预算约束。即使量子硬件出现不可预测的故障，这种架构也能确保查询结果值得信赖，使 Qute 在实际部署中兼顾积极探索与可靠性。

7 预备知识结果

原型实现。我们实现了 Qute 的一个最小原型，支持在 [1]中执行数据库过滤操作。经典执行引擎负责 SQL 解析、查询规划和结果协调，而数据过滤则交由量子计算后端处理。该原型使用 QPanda3 库（v1.0）实现，并在 origin_wukong 平台上开发。所有量子电路均在一台配备 72 个量子比特且存在噪声的真实量子处理器上执行。默认情况下，每次量子实验执行 2000 次测量，结果取平均值。

数据集。为评估量子计算在大规模数据上的有效性，我们采用随机生成方法构建了一个包含 $N = 2^{40}$ 个元组的合成数据集。我们考虑以下形式的过滤查询：SELECT RID FROM T WHERE $\phi(x)$，其中 ϕ 由针对生成属性的单属性谓词或合取谓词组成。谓词参数经过选择，以控制查询选择率；每个谓词的选择率均不超过 2%，并在此范围内生成多个选择率等级。

评估方法。鉴于经典数据库的运行时间对硬件和系统优化高度敏感，为公平比较算法复杂度，我们采用基于成本模型的评估。具体而言，我们评估两种方法：(1) 经典数据库基线，其性能使用传统成本模型估算 [28]；(2) Qute，在数据集规模允许时通过实际执行测量其性能，否则使用我们的成本模型（第 6 节）进行估算。

结果。 图5比较了数据规模不断增大时 Qute 与经典数据库的处理时间。对于 Qute，图中以离散样本点展示了小型数据集（最多 2^{10}）上的实测运行时间，Grover 误差为 ±8.0%；而成本模型估算的运行时间则以连续曲线表示。实测结果与估算结果高度吻合，表明成本模型能够准确反映 Qute 的性能，可可靠地用于大规模外推。此外，图中还报告了经典数据库的估算性能以供比较。结果显示，交叉点约为 $N \approx 2^{30}$。对于较小的数据集，经典数据库由于固定开销较低而运行更快；而对于较大的数据集，Qute 始终优于

经典方法。它突显了 Qute 在极大规模、无索引过滤工作负载中的可扩展性优势。

8 演进与未来方向

我们根据数据规模和计算能力，将量子数据库的演进划分为三个阶段（S1-S3）（图1）。❶ **量子辅助数据库**。量子计算充当经典系统的协处理器。由于量子比特数量有限、相干时间短且数据加载成本高，量子资源仅用于选择性、计算密集型任务，例如使用格罗弗搜索执行全表扫描。数据仍保存在经典内存中，并在运行时触发量子执行。此阶段没有索引加速；当量子运行失败或超过噪声阈值时，回退机制可确保正确性。主要瓶颈在于经典到量子的传输带宽。

❷ **量子中心型数据库**。随着量子存储器和加载方法的改进，量子计算成为主要

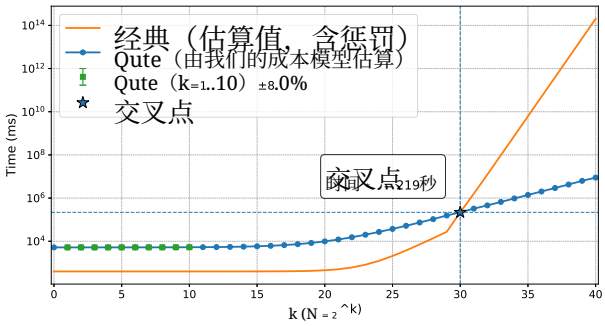


图5：性能比较（Qute 与 Classic）。

执行**量擎**-经典混合索引开始出现，使得能够选择性地访问叠加态。保真度感知优化器使用成本模型，在量子路径和经典路径之间进行规划，并将退相干、量子门延迟和量子比特数量限制纳入考量。路由机制据此分派子查询。连接和聚合等核心操作通过量子原语实现，包括 SWAP 测试和振幅估计，从而实现高效的混合执行。

❸ **量子原生数据库**。 在这一阶段，容错量子硬件能够支持更强大的存储和计算。所有数据类型（包括结构化、非结构化和多模态数据）均驻留在量子内存中。先进的数据编码（例如向量量化算法）可直接压缩和处理大型数据集。经典组件被降至最低，系统实现了完全量子原生的存储、索引和处理。

9 相关工作

面向数据库的量子计算。 先前的工作包括有关查询处理 [7, 19], 查询优化 [11, 12, 26, 31, 35, 38], 索引选择 [17, 22], 数据集成 [13], 和事务调度 [4, 5, 14]的综述与研究。其他研究则探索了量子辅助近似查询处理 [43] 和数据结构 [25]。尽管前景可期，这些工作大多仍保留经典数据模型和存储层，仅将量子计算用作孤立的加速步骤，未实现跨层集成。

面向量子计算的数据库。 多项工作探讨了经典数据库抽象如何支持量子计算。这包括在量子硬件上存储和操作关系数据 [24, 34], 将 SQL 查询编译为量子电路 [23]。量子机器学习也被应用于关系数据和图数据 [20, 21, 40, 42], 不过这些工作侧重于模型训练，而非数据库执行。

10 结论

本文展望了量子原生数据库将成为数据管理领域的一项基础性变革，使量子计算贯穿整个技术栈，从查询解析到执行。我们提出了 Qute，这是一个旨在应对算子编译、混合优化、选择性索引和保真度感知存储等关键挑战的系统。与以往基于模拟的方法不同，Qute 展示了一种统一的设计，支持实用的量子查询执行。我们还勾勒了三阶段路线图，概述了未来量子数据系统的架构里程碑和研究机会。

References

[1] Qute量子数据库原型）。https://github.com/weAIDB/Qute [2] Pablo Arrighi 和 Gilles Dowek。2012。物理丘奇 – 图灵论题与量子理论原理。《国际计算机科学基础杂志》23卷5期（2012），1131–1146页。[3] Martin Aumüller 和 Matteo Ceccarello。2022。使用局部敏感哈希实现分布式近似相似性连接。载于 EDBT：国际数据库技术拓展会议。[4] Tim Bittner 和 Sven Groppe。2020。通过使用量子退火调度事务来避免阻塞。载于第24届国际数据库工程与应用研讨会（IDEAS）论文集。ACM，1–10页。[5] Tim Bittner 和 Sven Groppe。2020。通过避免阻塞，利用量子退火加速事务调度方案的优化。《开放云计算期刊》7卷1期（2020），1–21页。[6] Gilles Brassard、Peter Høyer、Michele Mosca 和 Alain Tapp。2002。量子振幅放大与估计。《当代数学》305卷（2002），53–74页。[7] Umut Çalikyilmaz、Sven Groppe、Jinghua Groppe、Tobias Winker、Stefan Prestel、Farida Shagieva、Daanish Arya、Florian Preis 和 Le Gruenwald。2023。数据库量子加速的机遇：查询与事务调度方案优化。《VLDB基金会会刊》16卷9期（2023），2344–2353页。[8] Yanqi Chen、Xiao Yan、Alexandra Meliou 和 Eric Lo。2025。DiskJoin：使用固态硬盘的大规模向量相似性连接。《ACM SIGMOD》3卷6期（2025），1–27页。[9] Christoph Dürr 和 Peter Høyer。1996。一种寻找最小值的量子算法。载于第5届量子计算与量子通信国际会议（QCQC）论文集。施普林格，柏林、海德堡，362–371页。[10] Wang Fang 和 Qisheng Wang。2025。估计纯态保真度的最优量子算法。载于 ESA（LIPIcs，第351卷）。Schloss Dagstuhl - Leibniz 信息学中心，4:1–4:12页。[11] Tobias Fankhauser、Marc E. Solèr、Rudolf M. Füchslin 和 Kurt Stockinger。2021。采用经典计算与量子计算混合方法的多查询优化。arXiv 预印本 arXiv:2107.10508（2021）。https://arxiv.org/abs/2107.10508[12] Tobias Fankhauser、Marc E. Soler、Rudolf M. Füchslin 和 Kurt Stockinger。2023。使用基于量子门的量子计算机进行多查询优化。《IEEE Access》 11 2023 23066 23081 10 1109 2023 3254637 （），–。doi：。/ [13] ACCESS。。Kai Fritsch 和 Stefanie Scherzinger。。在量子计算机上求解数据库模式匹配的困难变体。《VLDB基金会会刊》16卷12期（2023），3990–3993页。[14] Sven Groppe 和 Jinghua Groppe。2021。通过为格罗弗搜索算法生成代码，在通用量子计算机上优化事务调度方案。载于第25届国际数据库工程与应用研讨会（IDEAS）论文集。ACM，149–156页。[15] Lov K. Grover。1996。一种用于数据库搜索的快速量子力学算法。载于 STOC。ACM，212–219页。[16] Lov K. Grover。1996。一种用于数据库搜索的快速量子力学算法。载于第28届 ACM 计算理论年度研讨会（STOC）论文集。212–219页。[17] Le Gruenwald、Rui Zhang 和 Jialu Zhou。2023。使用量子计算机和机器学习为大规模数据库应用进行索引调优。载于 VLDB 2023研讨会论文集。VLDB基金会。[18] Rihan Hai、Shih-Han Hung、Tim Coopmans、Tim Littau 和 Floris Geerts。2025。含噪声中等规模量子时代的量子数据管理。《VLDB基金会会刊》18卷6期（2025），1720–1729页。[19] Rihan Hai、Shih-Han Hung 和 Sebastian Feld。2024。量子数据管理：从理论到机遇。载于 ICDE。IEEE，5376–5381页。[20] Mohsen Heidari、Ananth Y. Grama 和 Wojciech Szpankowski。2022。迈向物理可实现的量子神经网络。载于美国人工智能促进协会（AAAI）。https://www.cs.purdue.edu/homes/spa/papers/AAAI21.pdf 量子神经网络的实现与训练方案。[21] Mohsen Heidari、Arun Padakandla 和 Wojciech Szpankowski。2021。量子数据学习的理论框架。载于 IEEE国际信息论研讨会（ISIT）。https://arxiv.org/abs/2107.06406 量子数据学习的理论基础。[22] Manish Kesarwani 和 Jayant R. Haritsa。2024。量子平台上的索引顾问。《VLDB基金会会刊》17卷12期（2024）。[23] Manish Kesarwani 和 Jayant R. Haritsa。2024。基于量子的 SQL 查询执行是否可行？载于 VLDB研讨会。VLDB.org。[24] Tuodu Li、Gongsheng Yuan、Chang Yao、Meng Shi、Ziyue Wang、Ling Qian 和 Jiaheng Lu。2024。关系数据库管理系统中表的量子存储设计。载于 VLDB 研讨会。VLDB.org。[25] Tim Littau、Ziyu Li 和 Rihan Hai。 [n. d.]。面向提升数据库性能的量子数据结构。VLDB 2024 研讨会 2150（[n. d.]），8097。[26] Hanwen Liu、Federico M. Spedalieri 和 Ibrahim Sabek。2025。Q^2O 的演示：量子增强型查询优化器。VLDB基金会会刊 18, 12 2025 5439 5443 10 14778 3750601 3750691 ( )，–。doi：。/。 [27] 2024 Hao Liu、Xiaotian You 和 Raymond Chi-Wing Wong。。首个树形量子数据结构：量子B+ 树。CoRR abs/2405.20416（2024）。[28] Stefan Manegold、Peter Boncz 和 Martin L Kersten。2002。面向分层存储系统的通用数据库成本模型。载于 VLDB'02：第28届超大型数据库国际会议论文集。Elsevier，191–202。[29] D. Michael Miller、Robert Wille 和 Zahra Sasanian。2011。多重控制托福利门的基本量子门实现。载于 ISMVL。IEEE Computer Society，288–293。[30] Mikko Möttönen、Juha J. Vartiainen、Ville Bergholm 和 Martti M. Salomaa。2005。使用均匀受控旋转变换量子态。量子信息与计算 5，6（2005），467–473。[31] Nitin Nayak、Jan Rehfeld、Tobias Winker、Benjamin Warnke、Umut Çalıkyılmaz 和 Sven Groppe。2023。通过在量子硬件和模拟器上求解QUBO问题来构建最优分叉连接树。载于新兴分布式环境中的大数据国际研讨会论文集（BiDEDE）。1–7。doi:10.1145/3579142.3594298[32] Michael A. Nielsen 和 Isaac L. Chuang。2016。量子计算与量子信息（10周年纪念版）。剑桥大学出版社。[33] Jessica Pointing、Oded Padon、Zhihao Jia、Henry Ma、Auguste Hirth、Jens Palsberg 和 Alex Aiken。2021。Quanto：通过自动生成电路恒等式优化量子电路。CoRR abs/2111.11387（2021）。[34] Carla Rieger、Michele Grossi、Gian Giacomo Guerreschi、Sofia Vallecorsa 和 Martin Werner。2024。量子数据库的操作框架。arXiv预印本 arXiv:2405.14947（2024）。https://arxiv.org/abs/2405.14947 定义了量子数据库框架中的量子数据存储与操作的相关运算及其实现。[35] Manuel Schönberger、Stefanie Scherzinger 和 Wolfgang Mauerer。2023。准备好跨越式发展（通过协同设计）了吗？量子硬件上的连接顺序优化。ACM数据管理会刊 1, 1 2023 1 27 10 1145 3581830 ( )，–。 [36] 2023 doi：。/ Prasanth Shyamsundar。。非布尔量子振幅放大与量子均值估计。量子信息处理 22，12（2023），423。[37] Yannick Stade、Lukas Burgholzer 和 Robert Wille。2025。迈向QIR支持：采用量子中间表示的步骤。载于 SC研讨会。ACM，1907–1915。[38] Immanuel Trummer 和 Christoph Koch。2016。在 D-Wave 2X绝热量子计算机上进行多查询优化。VLDB基金会会刊 9，9（2016），684–695。doi:10.14778/2947618.2947622[39] Nimish Ukey、Guangjian Zhang、Zhengyi Yang、Binghao Li、Wei Li 和 Wenjie Zhang。2023。针对动态高维数据的高效连续k近邻连接。万维网 26，6（2023），3759–3794。[40] Martin Vogrin、Rok Vogrin、Sven Groppe 和 Jinghua Groppe。2024。利用量子图神经网络对关系数据库进行监督学习。载于第二届量子数据科学与管理国际研讨会论文集（QDSM'24）。https://www.vldb.org/workshops/2024/proceedings/ QDSM/QDSM.5.pdf[41] Nathan Wiebe、Ashish Kapoor 和 Krysta M. Svore。2015。用于监督学习和无监督学习的近邻方法量子算法。量子信息与计算 15，3&4（2015），316–356。[42] Tobias Winker、Sven Groppe、Valter Uotila、Zhengtong Yan、Jiaheng Lu、Maja Franz 和 Wolfgang Mauerer。2023。量子机器学习：基础、新技术与数据库研究机遇。载于2023年数据管理国际会议（SIGMOD/PODS）伴随会议论文集。计算机协会，45–52。https://doi.org/10.1145/3555041.3589404[43] Sai Wu、Meng Shi、Dongxiang Zhang、Junbo Zhao、Gongsheng Yuan 和 Gang Chen。2024。当量子计算遇上数据库：用于近似查询处理的混合采样框架。IEEE知识与数据工程汇刊 36，12 2024 9532 9546（），–。[44] Jiadong Xie、Jeffrey Xu Yu 和 Yingfan Liu。2025。向量数据库中的快速近似相似连接。AC M SIGMOD 3，3，文章158（2025年6月），26页。doi:10.1145/3725403[45] Gongsheng Yuan、Yuxing Chen、Jiaheng Lu、Sai Wu、Zhiwei Ye、Ling Qian 和 Gang Chen。2024。面向数据库的量子计算：概述与挑战。CoRR abs/2405.12511（2024）。[46] Xinyu Zeng、Yulong Hui、Jiahong Shen、Andrew Pavlo、Wes McKinney 和 Huanchen Zhang。2023。列式存储格式的实证评估。《VLDB基金会会刊》17，2（2023），148–161。