Pneuma 项目：将信息需求具体化为关系模式，以自动化发现、指导准备并使数据与意图保持一致

Muhammad Imam Luthfi Balaka
luthfibalaka@uchicago.edu 芝加哥大学 芝加哥，美国

Raul Castro Fernandez
raulcf@uchicago.edu 芝加哥大学 芝加哥，美国

arXiv:2601.03618v1 [cs.DB] 7 Jan 2026

摘要

在数据管理生命周期中，数据发现和数据准备仍是长期存在的瓶颈，尤其是在用户意图模糊、不断变化或难以落地实施的情况下。Pneuma 项目推出了 Pneuma-Seeker，这是一个通过与由语言模型驱动的平台进行迭代交互，帮助用户表达并满足信息需求的系统。该系统将用户不断演化的信息需求具体化为关系数据模型，并逐步收敛为与该意图相一致的可用文档。为实现这一目标，系统结合了三项架构理念：通过上下文专门化减轻大语言模型在各子任务中的负担；采用 Conductor 式规划器组装动态执行计划；以及基于共享状态的收敛机制。该系统整合了检索增强生成（RAG）、智能体框架和结构化数据准备方面的最新进展，以支持半自动化、语言引导的工作流程。我们通过基于大语言模型的用户模拟对系统进行了评估，结果表明，该系统有助于揭示潜在意图、指导发现，并生成符合用途的文档。此外，它还可作为一种涌现式文档层，记录机构知识并支持组织记忆。

CCS概念

• **信息系统** → **数据管理系统**；**信息检索**； •**以人为中心的计算**；

关键词

数据发现、数据准备、信息需求、大型语言模型、人机交互

1 引言

大语言模型（LLM）有望支持整个数据管理生命周期，从数据收集到任务执行 [9, 18, 38,40]。在各个阶段中，最难实现自动化的环节包括数据发现，即识别并检索与用户信息需求 [4, 7], 相关的文档，以及数据准备，即将这些文档转化为可供下游任务使用的形式。一项关键挑战在于用户信息需求的本质：它往往模糊不清、不断变化，而且难以用软件能够有效执行的方式表达。随着大语言模型的能力日益增强，瓶颈正从执行任务转向帮助用户准确阐明自己的目标——也就是信息需求——以便利用现有数据满足这些需求。

请考虑一个来自我们大学财务部门的实际例子：“关税将对我们的组织产生什么影响？”回答这一问题需要完成以下几个步骤：

(1) 发现相关数据源，例如当前的关税税则和采购记录；(2) 定义“影响”，其中可能包括直接影响（例如来自征收关税的国家的进口商品）和间接影响（例如原本不受影响的进口商品中包含征收关税的零部件）；(3) 确定时间范围：用户关注的是下周、下一财政年度的预计影响，还是回顾性分析？(4) 将这些数据源集成为一份连贯的文档，以支持有意义的解读。

为了支持此类查询，系统必须提取、明确阐释并将问题背后的假设转化为可执行内容。这意味着要呈现“信息需求”，即回答用户问题所需的自然状态集合 [3]。明确阐述的信息需求随后会指导发现、集成和准备工作。在实践中，这些流程很少由单个用户独立完成；现代组织通常由业务分析师、数据科学家和数据工程师以半分散式方式协作，共同回答复杂问题 [36]。

Pneuma-Seeker 旨在帮助用户半自动地表达并满足此类信息需求。Pneuma-Seeker 的一个核心见解是，信息需求可以被具体化为数据模型，尤其是关系模式。随后，系统会尝试将该模式与可用数据对齐，并返回一份满足潜在信息需求的文档。起初，系统和用户都可能无法确切知道这份文档应是什么样子，但通过互动，双方会逐步完善各自的理解。例如，用户最初可能不会要求纳入直接关税和间接关税的影响，但在看到中间结果后，会明确提出这一要求。用户提供基于语言的反馈，而系统则提出越来越契合的数据模型，经过多轮迭代，最终逐渐收敛到满足预期信息需求的模型。

当用户向 Pneuma-Seeker 提出关税问题时，系统会检查包含数十个表格的采购数据库，确定其中缺少关税信息，然后从在线来源检索相关数据，将这些信息整合成表格结构，并提出一份初步文档供审核。经过两轮用户反馈后，系统最终确定了一个模式和一条 SQL 查询，用于估算德国供应商带来的关税影响，并与用户不断演变的“影响”理解相匹配。

本文根据知识共享署名 4.0 国际（CC-BY 4.0）许可协议发布。作者保留在其个人和公司网站上传播本文的权利，但须进行适当署名，即注明原始作品的作者和 CIDR 2026。第16届创新数据系统研究年会（CIDR ’26）。1月18日至21日，Chaminade，美国

在 Pneuma-Seeker 中，大语言模型充当语言与数据模型之间的桥梁：用户借助语言表达意图，而系统则利用数据模型引导发现和准备工作。Pneuma-Seeker 利用检索增强生成（RAG）[16] 和智能体架构的最新进展，使这座桥梁能够发挥实际作用。为实现这一功能，Pneuma-Seeker 提出了三项可泛化的贡献：

• **情境专门化**。大语言模型难以应对结构化数据任务中规模庞大且多样的上下文 [15, 17, 19]。Pneuma-Seeker 通过将工作流程拆分为具有专门上下文和明确范围的子任务来解决这一问题。

• **指挥者式规划**。Pneuma-Seeker 不采用静态的、基于规则的流水线，而是使用指挥者组件——一个由大语言模型驱动的智能体，根据实时证据动态制定计划，以推进信息需求的满足。

• **收敛标准**。Pneuma-Seeker 将不断演进的数据模型视为用户与系统之间的共享状态：用户通过语言对其进行完善，指挥者则利用它来指导下游模块。当文档与用户的潜在信息需求相一致时，便实现收敛。

为了在不涉及人类受试者的情况下评估 Pneuma-Seeker，我们使用 LLM Sim，这是一种基于大语言模型的智能体，扮演用户并仅对系统输出做出反应。虽然它无法完全替代真实研究，但定量和定性结果均表明，Pneuma-Seeker 有助于明确并满足信息需求。

Pneuma-Seeker 的设计带来了一个重要的副作用：它充当了组织知识捕获机制。通过促使用户阐明目标，它揭示了隐性信息需求和潜在数据流。在大规模应用下，这些交互会逐渐积累为涌现式文档，保存部落知识和组织记忆。随着团队不断发展，Pneuma-Seeker 让这些知识变得明确且可复用，从而增强内部数据生态系统的韧性。

2 从信息需求到答案

本节将定义相关概念，调研帮助用户满足这些需求的解决方案概览，并介绍一个促成 Pneuma-Seeker 设计的交互模型。我们旨在突出用户阐述和完善信息需求时反复出现的挑战，并提出一种结构化方法，使人类信息需求与系统表示共同演化，逐步趋近于潜在信息需求。

2.1 定义

信息需求。 信息需求是解决数据驱动任务所需的自然状态集合 [3]。这一定义具有普适性：信息需求可能包括训练分类器所需的特征、回答 SQL 查询所需的模式，或进行因果推断所需的变量。这些状态通常编码在文档中。为清晰起见，我们假设每项任务对应一个信息需求，但可能存在多个有效的表示形式。

潜在与主动信息需求。 潜在信息需求是完成任务所需的真实状态集合，用户起初往往并不知道这些状态。主动信息需求是用户对所需数据的工作假设，它会随着交互和探索不断演变，逐渐逼近潜在信息需求。

2.2 解决方案概览

文档检索。 一旦明确了主动信息需求，众多技术都可以帮助检索相关文档。例如，网络搜索引擎将关键词查询映射到匹配的文档。在结构化数据库中，SQL 查询会检索包含特定列的关系，前提是存在连接路径。检索增强生成系统支持大语言模型从向量存储或数据库中检索相关文本块，然后再组织答案。这些检索方法都假设信息需求已经得到较为清晰的表述，而这往往正是核心瓶颈 [2, 13, 33]。

识别信息需求。 另一个独立但至关重要的挑战，是帮助用户首先形成信息需求。许多用户界面提供支架来支持这一过程：网络自动补全功能利用群体层面先验；大语言模型接口通常会建议后续操作或澄清性提示；企业数据目录 [12] 会呈现使用模式（例如，查询表 A 和表 B 的用户通常也会使用表 C），帮助新用户形成对哪些数据相关的预期。

表示信息需求。 信息需求这一概念长期以来一直受到研究，尤其是在信息检索（IR）领域 [2, 27]。在信息检索中，信息需求通常被视为用户的潜在目标，而查询只能不完全地表达这一目标。在探索性搜索中，用户会进行开放式的调查性信息寻求，其目标会在搜索过程中不断演变 [22, 34]。类似地，在探索性数据分析（EDA）中，重点转向迭代完善和假设演变，这一过程通常由可视化引导 [11, 29]。在 Pneuma 项目中，我们在这些研究方向的基础上，为信息需求构建计算表示，使人类用户与机器能够共同演进。

在这项工作中，我们将信息需求具体化为一个关系数据模型，以及针对该模型的 SQL 查询。这一选择体现了我们的观点：解决数据驱动任务，最终需要实例化一份与用户意图一致的结构化文档（一张表或一组表格）。因此，系统的工作是识别能够实现潜在信息需求的模式和查询。

2.3 使数据与意图对齐：一种模型

我们提出以下模型来指导系统设计。人类用户希望完成一项数据驱动任务。该任务会引发一种潜在的信息需求；如果将这种需求表示为文档，该文档便足以完成任务。然而，用户一开始可能并不知道该文档是什么样的。

一个抽象系统起初并不专用于某项任务，而是维护着一个内部状态，用于表示其对信息需求不断演变的理解。交互以迭代方式进行：用户传达其主动信息需求，系统据此更新状态。在观察系统输出后，用户会修改输入，逐步引导系统更接近潜在的信息需求。当系统生成用户认为足够的文档（或模式 + 查询）时，交互便告结束。

在当今实践中，该系统并非只有软件——它是由人员角色和基础设施构成的复合体：业务分析师、数据科学家、数据工程师、领域专家、数据库、接口、API 以及更多内容 [36]。借助 Pneuma-Seeker，我们旨在将这些角色统一到一个连贯的软件系统中，通过迭代式的语言引导交互，帮助用户识别、阐述并实现其信息需求。

3 Pneuma-Seeker 系统

本节将介绍 Pneuma-Seeker，这是一个帮助用户识别并满足其潜在信息需求的系统。我们首先介绍其架构和关键设计原则，随后详细说明各个组件。

3.1 技术贡献与概述

Pneuma-Seeker 的设计基于三项技术洞见：

上下文专门化。 处理结构化数据时，大语言模型会受到上下文规模异质性的限制。上下文越异质，注意力就越分散在彼此无关的细节上。此外，使用不同的角色 提示大语言模型，有助于集中其行为 [28]。因此，Pneuma-Seeker 采用多组件架构，将检索、集成和编排分离。每个组件专注于一个子任务，大语言模型只接收与其指定角色相关的信息。

借助 Conductor 进行动态规划。 上下文专门化所产生的子任务，仍必须整合为一个端到端流程，以识别并满足用户的信息需求。Pneuma-Seeker 不依赖静态的基于规则的流水线，而是使用 Conductor（一种大语言模型驱动的智能体），根据满足信息需求的实时进展证据来编排流程。

用于收敛的共享状态。 Pneuma-Seeker 将用户的主动信息需求表示为关系数据模型（T，Q），其中 T是一组表格，而 Q 是针对 T的一系列 SQL 查询。用户与 Pneuma-Seeker 建立反馈回路：用户向系统描述不断变化的主动信息需求，系统更新 (T，Q)，用户再针对这些变化提供更多反馈。当用户停止交互，或主动信息需求与潜在信息相匹配时，交互便告结束需求（即收敛）。

这三项洞见直接塑造了 Pneuma-Seeker 的模块化架构。图1展示了该架构，它由多个组件组成：**(1) Conductor**，负责协调整个流程；**(2) 信息检索系统**，负责检索相关数据；以及**(3) Materializer**，负责整合并准备数据以形成 T。我们将在下文详细介绍这些组件。

3.2 Conductor

Conductor 通过实时选择操作，推动 Pneuma-Seeker 走向收敛，使其（T，Q）与用户的主动信息需求保持一致。目前，它一次只选择一个操作，但已有一些关于并行化大语言模型规划的研究（例如，[37, 41]）。在 Conductor 中，操作可以是以下任意一种：

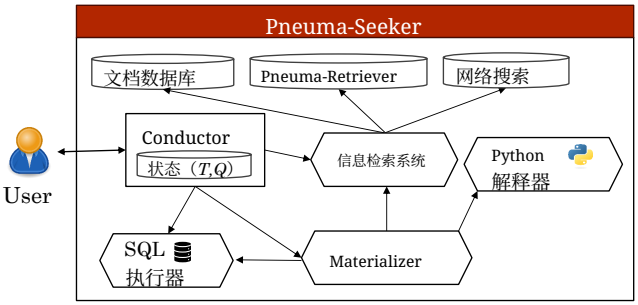


图1：Pneuma-Seeker 的架构

- **内部推理。** Conductor 会进行内部推理（受 ReAct [35] 启发），我们通过提示大语言模型，使其能够评估当前状态（T，Q）、从信息检索系统检索到的数据，以及用户最近的反馈，从而决定接下来最合适的操作。例如，假设用户澄清说，要确定新关税的影响，就需要知道之前生效的关税。Conductor 可能会这样推理："我当前的查询仅使用新关税的百分比来计算价格变化。我应该检索之前的关税百分比，并将最终百分比更新为 （新_关税 – 之前的_关税）。"
- **工具调用。** Conductor 可以直接调用工具，包括信息检索系统、Materializer 和 SQL 执行器，也可以先通过内部推理识别需求，再调用这些工具。例如，如果它判断需要先前生效的关税，就可以向信息检索系统发出检索请求："检索该地区先前生效的关税。"同样，如果它决定应当将 T物化，就可以调用 Materializer，并附上说明，例如："物化 T，这是一个包含相关采购数据的表格，其中列有该供应商所在国家的新关税信息和先前生效的关税信息。"
- **状态修改。** Conductor 可以更新状态（T，Q），以修订对用户主动信息需求的解读。它可以仅修改 T、仅修改 Q，或同时修改两者。例如，在确定 Q 应考虑之前的主动关税后，它可以更新 Q，使其变为：*["SELECT price * (1 + new_tariff - previous_tariff) AS new_price FROM procurement_data"]*。
- **面向用户的沟通。** Conductor 可以直接与用户互动，解释所采取的操作、总结当前状态、提出澄清性问题，或建议后续步骤。例如，在执行 Q 中的查询后，它可以告知用户："我已执行 Q 中的查询。从德国供应商 *12345* 处购买的商品，其新价格上涨了 *5%*，平均涨幅为 *5,125* 美元。"

按设计，Conductor 不受固定流程规则约束，但必须遵守必要的依赖关系（例如，必须先将 T物化，才能执行 Q）。Conductor 将连续操作的次数限制为固定值 i。根据经验观察，我们将该值设为 $i = 5$，因为在评估过程中，大语言模型很少达到这一上限。此限制旨在防止（T，Q）在用户反馈能够纠正之前偏离潜在信息需求，同时避免长时间自主运行，让用户久候。此外，我们还指示 Conductor 在每一组操作结束时，

尽可能向用户发送消息。如果达到操作次数限制时仍未生成面向用户的消息，系统就会中断并强制 Conductor 发送此类消息。

与用户交互时，Conductor 会依据从信息检索系统中检索到的数据做出决策，而不是仅凭假设行事（例如，假设采购表中存在 ID 为 12345 的供应商）。这样可以避免生成无法实现的不切实际的（T，Q）状态。例如，如果用户要求分析 2019 年针对某个特定供应商的新关税所产生的影响，但信息检索系统只检索到 2020 年及之后的关税记录，Conductor 就能发现这一缺口，并寻找替代来源或通知用户，而不是耗费多次行动来构建并实现（T，Q）。这可以避免白费功夫，并使交互聚焦于可实现的目标。

在整个交互过程中，Pneuma-Seeker 不仅会向用户展示面向用户的回复，还会展示当前状态（T，Q）。图 2 展示了该界面（截至 2025年8月）。将状态（框 3；T展示了示例行）与聊天界面（框 1 和框 2）并列显示，用户便能发现其意图与系统不断演进的模型之间细微的不匹配，并加以纠正。继续以关税影响为例，假设 T 包含采购数据，但完全没有国家属性，尽管用户只关心德国供应商。缺少这一属性，Q 就无法正确筛选数据，最终计算结果也会偏离用户的意图。

3.3 信息检索系统

信息检索系统通过从多个来源检索相关数据，为 Conductor 和 Materializer 提供支持。它将表格和文本等异构检索格式抽象为文档对象。这种统一表示方式使我们能够添加新的检索器，而无需更改信息检索系统其余部分的设计。尽管如此，Conductor 和 Materializer 都了解系统中可用的数据类型：目前包括表格、领域知识和网页，这些数据在当前实现中由以下三个检索器处理：

• **Pneuma-Retriever [1]**, 一种先进的表格发现系统，采用混合索引，将基于 HNSW [21] 的向量存储与基于 BM25 [26] 的倒排索引相结合，以实现高效的表格搜索。

• **文档数据库**，它使用 Pneuma-Retriever 的索引器来存储领域知识，从而实现跨用户知识迁移。例如，如果一位用户指出，估算关税影响时需要同时考虑直接关税和间接关税，那么后续与关税相关的查询就可以利用这一见解。受先前研究（例如 [20, 23, 30]）的启发，Pneuma-Seeker 会自动从用户交互中捕获知识，并将其保存到文档数据库中。

• **网络搜索**，它为外部搜索引擎提供了一个轻量级接口，用于查询一般信息或最新信息。

按照当前实现，Pneuma-Seeker 已经能够处理结合结构化和非结构化数据源的复杂用户信息需求。在此设计基础上，我们将扩展信息检索系统，使其能够接收来自 Conductor 或 Materializer 对检索到的文档的直接反馈。

3.4 Materializer

Materializer 的唯一目的是向 T 填充数据，这可能涉及信息检索系统对多源数据的集成。Materializer 与 Conductor 相互独立，体现了 **上下文专门化** 原则：Materializer 仅处理与数据集成和转换相关的上下文，不会被编排细节分散注意力。Materializer 还会考虑 Q，因此能够理解查询中的筛选条件。例如，如果 Q中的查询要求日期列采用“yyyy-mm-dd”格式，而该列实际列出的值采用“Month Day, Year”格式，Materializer 就会通过生成 Python 代码来转换该列的值。

Materializer 的工具包包括一个基于 DuckDB [25] 的 SQL 执行器，用于执行标准关系运算（连接、并集等），以及一个配备 Pandas [32] 和 NumPy [10] 的 Python 解释器。与信息检索系统类似，Materializer 采用可扩展设计，可以轻松集成新的运算符，包括语义关系运算（例如 LOTUS [24] 定义的运算）。

通过提示词，Materializer 能够获知从信息检索系统检索到的所有文档，并利用这些文档生成 SQL 查询或 Python 代码。例如，如果 Materializer 检索到两份文档，分别代表（1）通过网络搜索获取的最新关税信息，以及（2）来自 Pneuma-Retriever 的内部采购表格，那么它就可以生成 Python 代码来合并这两份文档（例如，将关税信息作为表格中的新列）。过程中可能会出现错误（例如，对包含空值的列执行浮点运算），因此相应的工具会分析这些错误，并向 Materializer 提供反馈，以修正生成的查询或代码。

我们计划探索这样的交互模型：Conductor 在每次 n 操作后向 Materializer 提供增量反馈，以此模拟用户与 Conductor 的交互方式。这种额外的监督有助于防止 Materializer 偏离预期的 T，尤其是在较长的集成流水线中。

3.5 动态流水线与静态流水线

本节将回顾我们尝试过的早期设计，并根据这些经验说明 Pneuma-Seeker 设计的合理性。

与我们最初尝试的静态流水线相比，我们设计的 Conductor 和 Materializer 更易于扩展，以纳入更多操作/工具（从而覆盖更多用例）——定义 (T，Q)、检索排名前-k 的表格、通过关系运算筛选并集成这些表格，再将集成结果精简至 T 中定义的内容。虽然这样的静态流水线能够覆盖各种用例，但随着纳入的操作/工具越来越多，更新起来也会更加困难。每当原有流水线中的某个步骤无法满足需求时（例如，缺少覆盖新用例所需的操作），就必须重新审视整个流水线。

例如，在我们的关税影响场景中，正确的分析可能需要将采购数据与关税记录连接起来，但这些数据源并不共享通用标识符（例如 supplier_id）。在这种情况下，系统需要采用更灵活的关系运算（例如语义连接或模糊连接），而这些运算并不属于原有流水线的一部分。将此类运算纳入静态、预定义的流水线可能需要大量工程工作，因为必须谨慎选择它在流水线中的位置

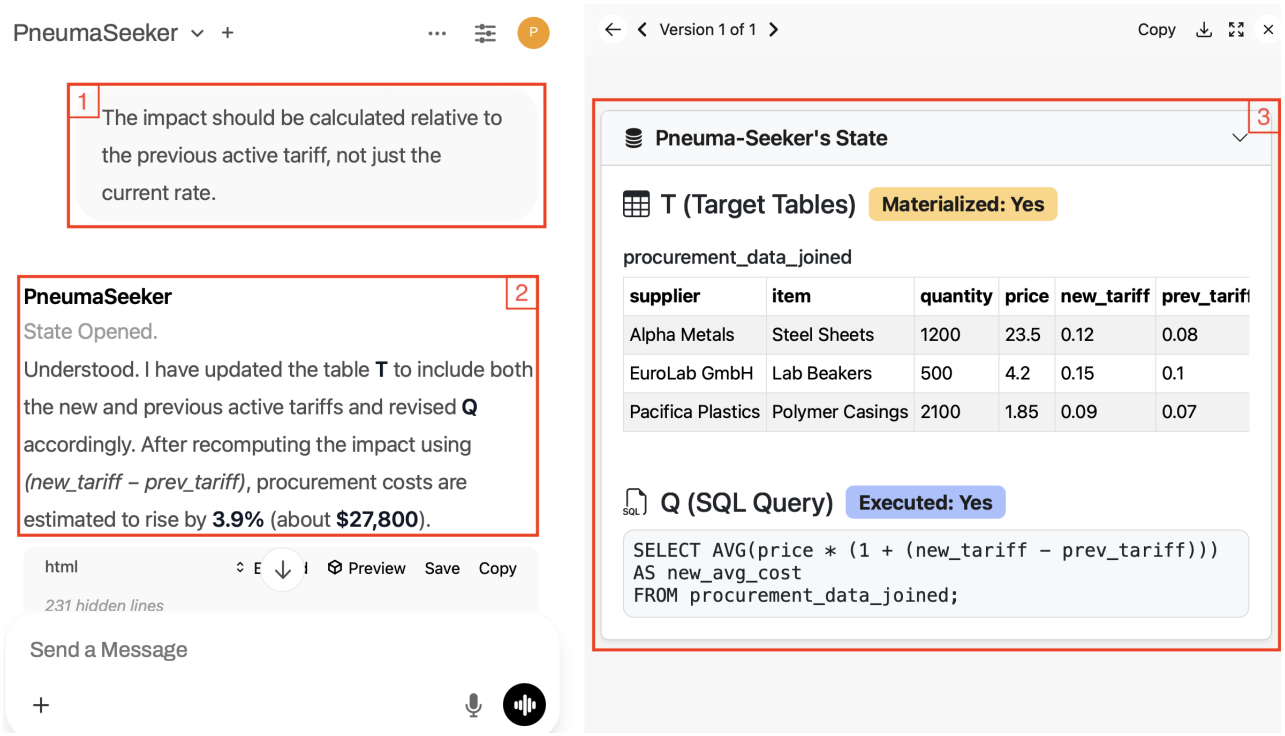


图 2：Pneuma-Seeker 界面，展示了：[1] 用户查询（澄清）、[2]面向用户的消息，以及 [3] 状态视图页面（T，Q）。注：出于隐私考虑，此处显示的 T数字和值并非真实数据。

，并确保其假设与上游和下游运算都相符。

换言之，可以将静态流水线泛化以支持各种用例，但每次扩展都需要投入越来越多的工作。随着添加更多操作/工具，这些集成成本会不断叠加。相比之下，使用 Conductoror Ma- terializer 时，我们只需定义新的操作或工具，它们便能自然地融入其操作空间。

3.6 Pneuma-Seeker 实战演示

为了说明各组件如何协同工作，我们重新审视第 1 节中财务部门提出的宽泛问题："关税会对我们的组织产生什么影响？"

在交互过程中，用户补充了一项关键澄清："影响应相对于此前生效的关税计算，而不仅仅是依据当前税率。"

Conductor 首先进行**内部推理**，意识到要准确计算影响，需要同时获取最新和此前生效的关税信息。随后，它向**信息检索系统**发出工具调用，系统从 Pneuma-Retriever 返回采购表，并从网络搜索界面返回关税信息。基于这些信息，Conductor 将 T 定义为一张合并采购数据与关税信息的表格。它总结检索到的来源并询问用户："我找到了相关供应商的采购记录和历史关税信息。我已设计好 T，用于将关税信息与采购数据合并。现在要将 T物化吗？"

用户确认后，Conductor 调用**Materializer**合并采购数据与关税数据，从而填充 T。T物化完成后，Conductor 执行**状态修改**，更新Q，使其根据此前的关税计算成本影响：

["SELECT AVG(price * (1 + (new_关税 - prev_关税))) AS new_avg_cost FROM procurement_数据_joined"]

SQL 执行器运行更新后的查询，并返回所有商品的采购成本预计增加 x%（约 $y）。[1] Con- ductor 通过**面向用户的通信**向用户报告这一信息：

> "如果新关税生效，预计采购成本将上升 x%（约 $y），这一估算是相对于此前的税率计算的。"

此时，用户可以进一步缩小范围，例如仅考虑实验室设备和来自德国的供应商，从而触发另一个细化周期。

4 评估

在本节中，我们通过回答以下问题，初步展示 Pneuma-Seeker 帮助用户阐明信息需求并完成数据任务的能力：

- **研究问题1**：用户能否通过与 Pneuma-Seeker 交互，找到自己内在的信息需求？
- **RQ2**：给定特定的信息需求，Pneuma-Seeker 能否准确地满足该需求？

[1]出于隐私原因，实际数字不予公开。

评估工作负载。 我们在 KramaBench [14], 中的考古学和环境数据集上评估 Pneuma-Seeker，并禁用网络搜索，以防止从互联网泄露基准测试信息。这些数据集分别对应 12 个和 20 个问题。表1展示了这些数据集的特征。

表1：数据集的特征

数据集	# 表格	平均行数	平均列数
考古学	5	11,289	16
环境	36	9,199	10

每个基准问题都是一个潜在信息需求。我们使用大语言模型（GPT-4o）来模拟一位领域专家（LLM Sim）与系统进行交互。从一个宽泛的提示开始，LLM Sim 根据系统的输出，逐步细化其主动信息需求。例如，给定以下潜在问题：

"研究首次和最后一次记录马耳他地区人群时，钾的平均值是多少（百万分率）？假设钾在样本之间呈线性插值。请将答案四舍五入到小数点后 4 位。"

初始查询为：

"我很想深入了解马耳他地区的历史数据。你能帮我概览一下过去的研究涉及哪些不同变量吗？"

重要的是，收敛并非必然：LLM Sim 的主动信息需求可能永远无法与潜在信息需求完全匹配。作为参考，我们在图3中展示了提供给 LLM Sim 的提示。

你正在模拟*{domain_专家_描述}*，该专家正与一个数据发现系统交互，以从企业数据集中探索洞见。

*{*根据系统：
- *Pneuma-Seeker*：系统会将你的信息需求表示为一组目标模式和 *SQL* 语句；执行这些语句即可获得答案。它可以组合、转换和推理数据，帮助你进行探索。

- *FTS* 或 *Pneuma-Retriever*：系统只会根据你的描述返回相关表格。它不会推断你更深层的意图，也不会组合或分析数据。*}*

场景：
- 系统已能访问内部数据集。- 你（模拟用户）熟悉该领域，并见过类似的数据集。- 你并非要上传新数据集，也不是在询问这些数据集是否存在——你假定它们确实存在。

可能的最终目标（开始时未知）：
{question}

行为：- 根据系统的响应，逐步探索并完善你的问题。- 可以含糊其辞或探索旁支，就像一名好奇的分析师一样。- 只有当系统的输出正确地引导你到上述具体问题时，才提出该问题。

继续扮演领域专家。这是目前为止的对话（请像直接向系统发出提示一样作答）：你：*{initial_broad_prompt}*

图3：用于 LLM_Sim 的提示

4.1 研究问题1：收敛

收敛是指 LLM Sim 的主动信息需求与其潜在信息需求相匹配。我们引入两项指标：**（1）收敛比例**，即 LLM Sim 实现收敛的基准问题所占比例；以及**（2）收敛中位轮数**，即 LLM Sim 必须向系统发出提示以实现收敛的次数中位数（上限设为 15 次）。

我们将 Pneuma-Seeker 与三个基线进行比较：基于 BM25 的全文搜索（FTS）、Pneuma-Retriever 和 LlamaIndex[2]（一个具有代表性的检索增强生成系统）。FTS 和 Pneuma-Retriever 是静态系统，只返回表格，并以其列和样本行作为表示。LlamaIndex 则在 top-k 向量检索器之上加入大语言模型，以便为 LLM Sim 解读检索到的数据。

如图4和图5所示，结果表明，Pneuma-Seeker 始终实现最高的收敛比例，并且收敛中位轮数与 LlamaIndex 相近。Pneuma-Seeker 和 LlamaIndex 不仅能够呈现相关数据，还能对其进行解读，将响应置于具体语境中并反馈给 LLM Sim。尽管 LLM Sim 最初提出的查询更为宽泛，这些系统仍会呈现相关数据和建议，并将其传达给 LLM Sim。随后，LLM Sim 会响应这些系统，以进一步探索并逐步接近揭示其潜在信息需求，而系统也会相应调整（例如修改状态）。

[2]https://www.llamaindex.ai/

另一方面，由于 LLM Sim 必须自行解读结果，FTS 和 Pneuma-Retriever 都难以收敛。此外，尽管 Pneuma-Retriever 几乎能够在所有问题中找到正确的表格，LLM Sim 只能查看示例行，以避免触及上下文限制。即使 GPT-4o 具有 128k 的上下文限制，使用我们的数据集进行 2～3 轮交互也足以超出该限制。在大多数问题中，LLM Sim 不断尝试调整其查询，以便从检索到的表格中获取更具体的信息。

Pneuma-Seeker 与其他系统之间存在延迟权衡。平均而言，Pneuma-Seeker 响应提示需要 70.26 秒，而 FTS 和 Pneuma-Retriever 几乎可以即时响应。此外，Pneuma-Seeker 使用的大语言模型 OpenAI 的 O4-mini，每 100 万个输入词元和输出词元分别产生 1.1 美元和 4.4 美元的费用。作为参考，我们在表 2 中列出了 LLM Sim 与 Pneuma-Seeker 使用多种模型交互的估算平均成本。

4.2 RQ2：准确率

仅仅收敛到潜在信息需求还不够；用户最终希望获得针对其信息需求的准确答案。

表 2：不同大语言模型的平均令牌使用量和成本估算

数据集	平均输入词元	平均输出词元	Haiku 4.5		O4-mini		O3		gpt-5.1		Sonnet 4.5		Opus 4.5	
			In	Out	In	Out	In	Out	In	Out	In	Out	In	Out
考古学	248,351	2,854	$0.25	$0.01	0.27 美元	0.01 美元	0.50 美元	0.02 美元	0.31 美元	0.03 美元	$1.49	$0.04	$1.24	$0.07
环境	149,011	1,712	$0.15	$ 0.01	$0.16	$0.01	$0.30	$ 0.01	$0.19	$0.02	$0.45	$0.03	$0.75	$0.04

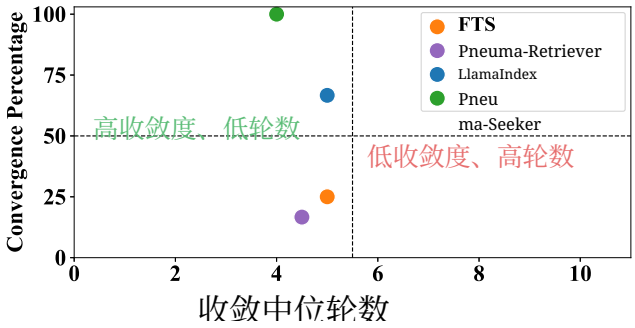


图4：收敛中位轮数与收敛百分比的比较（考古学数据集）

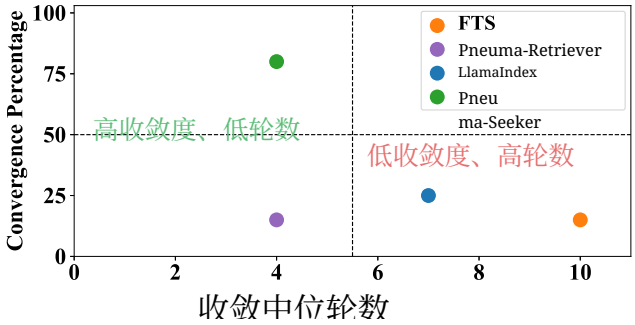


图5：环境数据集的收敛中位轮数与收敛百分比对比

FTS 和 Pneuma-Retriever 并非为提供答案而设计，因此我们在 RQ2 中排除它们，并纳入新的基线：DS-Guru 和 OpenAI 的 O3。DS-Guru 是 Kramabench 的参考框架，它指示大语言模型将问题分解为一系列子任务，逐步推理，并综合生成 Python 代码来执行该计划。我们选择基于O3的DS-Guru，因为它的表现最佳。

O3 是目前最优秀的推理模型之一，上下文限制为 200k。对于每个基准问题，我们都会提供所有相关表格，因此它拥有回答问题所需的全部信息。然而，在考古学问题中，12 题有 6 题、环境问题中，20 题有 17 题，O3 都出现了上下文长度超限错误。O3 未能正确回答 6 道考古学问题中的任何一道，但正确回答了 2 道环境问题。总的来说，提供所有相关上下文仍然不是一种可扩展的方法。

其余系统的结果如表3所示。Pneuma-Seeker 在所有数据集上都优于其他系统。尽管 Pneuma-Seeker 使用的模型更小、成本效益更高，但其表现仍优于 DS-Guru。另一方面，LlamaIndex 未能正确回答任何问题，因为这些问题需要实际计算（例如，计算某一列的平均值），而不只是解读排名前-k 的上下文。

表3：各数据集准确率对比

系统	考古学	环境
LlamaIndex	0.00%	0.00%
DS-Guru (O3)	25.00%	19.60%
Pneuma-Seeker	**41.67%**	**55.00%**

5 讨论

在本节中，我们将讨论构建 Pneuma-Seeker 的经验教训，并阐述我们对 Pneuma 项目的愿景。

5.1 经验教训。

过去一年中，我们构建了 Pneuma-Seeker。我们总结出以下经验教训，这些经验共同促成了本文所述架构的形成，并直接影响了各个组件及其交互方式的设计。

动态流水线具有更强的适应性。我们的早期原型遵循固定的处理序列。当然，我们可以调整这条流水线，加入更多操作/工具，从而覆盖更多用例。然而，我们意识到，随着流水线不断演进，所需投入的精力也会越来越多。一旦遇到系统尚未覆盖的用例，例如需要新增操作或交互性，我们就不得不进行大量重新设计。

这种情况屡屡发生。例如，我们必须确定哪些操作是必需的、哪些是可选的，是否跳过或重新启动某些操作，以及如何在不影响现有流水线稳定性的前提下集成新操作。静态方法可以适应这些情况，但随着系统纳入越来越多的操作和功能，工程负担会迅速增加。

动态流水线也并非万能药：如果缺少必要的能力（例如语义连接），系统仍然会失败。然而，关键区别在于，动态框架中缺失的能力是局部化的：一旦实现，这些能力就能自然地嵌入系统的操作空间。相比之下，将相同的能力添加到静态流水线中，则必须重新审视整个流水线。

这一认识与人工智能赋能系统的更广泛理念相一致：系统不应将僵化的序列硬编码，而应根据不断变化的状态、可用工具和积累的领域知识，动态地选择操作。这也呼应了数据处理的声明式理念：指定需要什么，而不是如何实现。

现有系统（如 ReAcTable [39] 和Chain-of-Table [31]）也在更具体的场景中采用动态编排：假设处于传统的表格问答场景，且事先已知一张特定的表格。而在我们的场景中，系统可能需要多个表格和非表格数据，才能满足用户的信息需求。总体而言，处理用户反馈、维持状态的迭代对齐，以及整合异构数据源等因素，都强烈地不利于采用静态流水线。这些观察共同促使我们放弃预定义流水线，转而采用 Conductor 灵活的、由状态驱动的动态编排。

上下文专门化至关重要。 动态系统的一项关键挑战在于组件本身的设计。我们观察到，在不同组件（例如 Conductor 和 Materializer）之间进行上下文专门化十分重要。除了降低幻觉发生的可能性（因为每个组件都专注于自身的狭窄职责）之外，专门化还有助于避免超出上下文窗口限制；当大语言模型同时被提示承担多个不同角色时，更容易触及这一限制。

展望未来，尽管目前已有大量关于扩展上下文窗口的研究，未来的模型也很可能支持更大的窗口，但即使所有输入词元都能放入窗口，也无法保证模型会有效关注（或有意义地利用）它们。与仅仅三年前相比，当前模型的上下文窗口已经长得多，但近期研究仍表明，随着上下文长度增加，大语言模型在问答等任务上的表现会下降，即使相关信息完全可以检索到也是如此 [6]。 [5] 等理论分析也表明，序列长度越长，模型聚焦特定词元的能力就越弱。因此，我们认为上下文专门化仍然有益，尤其是随着系统不断扩展（例如新增操作和工具）时。

模式优先表示有助于实现对齐。 将信息需求具体化为关系模型，为用户和系统提供了共同的依据，使双方能够据此协同推理。它为用户提供了一个具体对象，可用于核验结果，而不必完全依赖面向用户的自然语言消息。状态（T，Q）由此成为一种结构，可以反复完善、纠正或扩展，从而减少沟通误解。

呈现中间状态（T，Q）能让用户（以及 LLM Sim）及早发现细微的误解，避免其影响后续步骤。仅靠自然语言并不足够；用户需要了解不断演变的数据模型，才能提供具体且有意义的反馈。这也促使用户具体思考自己究竟需要什么。如何有效传达这一不断演变的状态（以及研究用户如何理解并据此行动），仍是未来用户研究的重要方向。

5.2 我们的愿景

我们在一年半前启动了 Pneuma 项目，此前已在数据发现领域开展了五年多的研究。我们的第一个里程碑是 Pneuma-Retriever [1]。本文介绍了该项目当前的状态。此后，项目不断发展，旨在解决以数据为中心的组织面临的一个关键瓶颈：随着大语言模型的能力日益增强，并越来越深入地融入数据工作流，主要挑战转变为帮助用户清晰表达自己想从数据中获取什么。我们的愿景是构建一个将这一表达过程视为首要任务的系统。我们的设计体现了这一愿景：将信息需求具体化为关系模式，并根据用户当前的意图量身定制、即时构建。

我们着重指出 Pneuma-Seeker 设计所带来的一项重要新兴效应。通过提示用户清晰表达目标、披露假设并外化其心智模型——这种行为在大语言模型驱动的界面中日益常见——Pneuma-Seeker 得以捕捉那些通常属于隐性知识的内容。在组织内部，这种“部落知识”代表着集体智慧储备。如果能让其可搜索且持久保存，就有望实现我们在早期工作 [8] 中提出的内部数据市场愿景。此类市场将使组织能够从其数据资产中挖掘出远超以往的价值，远不止于本文所探讨的数据发现与准备所带来的直接收益。过去几年里，我们不断推进这一愿景，但挖掘部落知识仍是一道难以攻克的障碍。Pneuma 项目是我们迄今为止为消除这一障碍所做的最新、最有针对性的尝试，并借此释放组织数据中潜藏的价值。

参考文献

[1] Muhammad Imam Luthfi Balaka、David Alexander、Qiming Wang、Yue Gong、Adila Krisnadhi 和 Raul Castro Fernandez。Pneuma：利用大语言模型在端到端系统中实现表格数据表示与检索。*Proc. ACM Manag. Data*，3(3)，2025 年 6 月。doi：10.1145/3725337。URL https://doi.org/10.1145/3725337。[2] N.J. Belkin。将异常知识状态作为信息检索的基础。*Canadian Journal of Information Science*，5(1)：133–143，1980 年。[3] Raul Castro Fernandez。数据的价值是什么？一种理论与系统化方法。*ACM / IMS J. Data Sci.*，2(1)，2025 年 6 月。doi：10.1145/3728476。URL https://doi.org/10.1145/3728476。[4] Raul Castro Fernandez、Ziawasch Abedjan、Famien Koko、Gina Yuan、Samuel Madden 和 Michael Stonebraker。Aurum：一个数据发现系统。载于*2018 IEEE* 第 *34*1012，2018 1001 10 1109 2018 00094届数据工程国际会议（*ICDE*），页码 – 。doi：。/ICDE。 。[5] Shi Chen、Zhengjiang Lin、Yury Polyanskiy 和 P hilippe Rigollet。长上下文 Transformer 中的临界注意力缩放，2025 年。URL https://arxiv.org/abs/2510。05554。[6] Yufeng Du、Minyang Tian、Srikanth Ronanki、Subendhu Rongali、Sravan Babu Bodapati、Aram Galstyan、Azton Wells、Roy Schwartz、Eliu A Huerta 和 Hao Peng。尽管检索完美，仅上下文长度也会损害大语言模型性能。载于 Christos Christodoulopoulos、Tanmoy Chakraborty、Carolyn Rose 和 Violet Peng 编，计算语言学协会论文集：*EMNLP 2025*，页码 23281–23298，中国苏州，2025 年 11 月。计算语言学协会。ISBN 979-8-89176-335-7。doi：10.18653/v1/2025.findings-emnlp.1264。URL https://aclanthology.org/2025.findings-emnlp.1264/。[7] Raul Castro Fernandez。数据发现是一个社会技术问题：从文档识别与检索到数据生态的路径。载于*IEEE* 计算机学会数据工程简报，2025 年。URL https://api.semanticscholar.org/CorpusID: 281957015。预印本可在 Semantic Scholar（CorpusID:281957015）获取。[8] Raul Castro Fernandez、Pranav Subramaniam 和 Michael J Franklin。数据市场平台：交易数据资产以解决数据问题。*Proceedings of the VLDB Endowment*，13(11)。[9] Raul Castro Fernandez、Aaron J. Elmore、Michael J. Franklin、Sanjay Krishnan 和 Chenhao Tan。大语言模型将如何颠覆数据管理。*Proc. VLDB Endow.*，16(11)：3302–3309，2023 年 7 月。ISSN 2150-8097。doi：10.14778/ 3611479.3611527。URL https://doi.org/10.14778/3611479.3611527。[10] Charles R. Harris、K. Jarrod Millman、Stéfan J. van der Walt、Ralf Gommers、Pauli Virtanen、David Cournapeau、Eric Wieser、Julian Taylor、Sebastian Berg、Nathaniel J. Smith、Robert Kern、Matti Picus、Stephan Hoyer、Marten H. van Kerkwijk、Matthew Brett、Allan Haldane、Jaime Fernández del Río、Mark Wiebe、Pearu Peterson、Pierre Gérard-Marchant、Kevin Sheppard、Tyler Reddy、Warren Weckesser、Hameer Abbasi、Christoph Gohlke 和 Travis E. Oliphant。使用 NumPy 进行数组编程。《自然》，585(7825)：357–362，2020 年 9 月。ISSN 1476-4687。

doi：10.1038/s41586-020-2649-2。URL https://doi.org/10.1038/s41586-020-2649-2。[11] Jeffrey Heer 和 Ben S hneiderman。用于可视化分析的交互动态。计算机协会通讯，55(4)：45–54，2012 年 4 月。doi：10.1145/2133806.2133821。[12] Nils Jahnke 和 Boris Otto。企业中的数据目录：应用与集成。*Datenbank Spektrum*，23：89–96，2023 年。doi：10.1007/s13222-023-00445- 2。[13] Carol C. Kuhlthau。搜索过程探析：从用户视角看信息寻求。*Journal of the American Society for InformationScience*，42 5 361 371,( )：–。doi：。/1991 10 1002 1097 4571 199106 42 5<361 6>3 0 2 (SICI) - ( ) : ::AID-ASI 。.CO; -#。[14] Eugenie Lai、Gerardo Vitagliano、Ziyu Zhang、Sivaprasad Sudhir、Om Chabra、Anna Zeng、Anton A. Zabreyko、Chenning Li、Ferdinand Kossmann、Jialin Ding、Jun Chen、Markos Markakis、Matthew Russo、Weiyang Wang、Ziniu Wu、Michael J. Cafarella、Lei Cao、Samuel Madden 和 Tim Kraska。Kramabench：面向数据湖上数据到洞察管道的人工智能系统基准测试。*ArXiv*，abs/2506 06541，2025 279250249。。URL https://api.semanticscholar.org/CorpusID: 。[15] Younghun Lee、Sungchul Kim、Ryan A. Rossi、Tong Yu 和 Xiang Chen。学习约简：旨在提升大语言模型在结构化数据上性能的方法，2024 年。URL https://arxiv.org/abs/2407.02750。[16] Patrick Lewis、Ethan Perez、Aleksandra Piktus、Fabio Petroni、Vladimir Karpukhin、Naman Goyal、Heinrich Küttler、Mike Lewis、Wen-tau Yih、Tim Rocktäschel、Sebastian Riedel 和 Douwe Kiela。面向知识密集型 NLP 任务的检索增强生成。载于神经信息处理系统第三十四届国际会议论文集，NIPS '20，美国纽约州雷德胡克，2020 年。Curran Associates Inc. ISBN 9781713829546。[17] Liyao Li、Jiaming Tian、Hao Chen、Wentao Ye、Chao Ye、Haobo Wang、Ningtao Wang、Xing Fu、Gang Chen 和 Junbo Zhao。LongTableBench：跨现实世界格式与领域的长上下文表格推理基准测试。载于 Christos Christodoulopoulos、Tanmoy Chakraborty、Carolyn Rose 和 Violet Peng 编，计算语言学协会论文集：*EMNLP 2025*，页码 11927–11965，中国苏州，2025 年 11 月。计算语言学协会。ISBN 979-8-89176-335-7。doi：10.18653/v1/2025.findings-emnlp.638。URL https://aclanthology.org/2025.findings-emnlp.638/。[18] Lei Liu、So Hasegawa、Shailaja Keyur Sampat、Maria Xenochristou、Wei-Peng Chen、Takashi Kato、Taisei Kakibuchi 和 Tatsuya Asai。AutoDW：利用大语言模型实现自动数据整理。载于第 *39* 届 *IEEE/ACM* 自动化软件工程国际会议论文集，ASE '24，美国纽约州纽约市，2024 年，第 2041–2052 页。计算机协会。ISBN 9798400712487。doi：10.1145/3691620.3695267。URL https: //doi.org/10.1145/3691620.3695267。[19] Tianyang Liu、Fei Wang 和 Muhao Chen。使用大语言模型重新思考表格数据理解。载于 Kevin Duh、Helena Gomez 和 Steven Bethard 编，*2024* 年北美计算语言学协会分会会议论文集：人类语言技术（第 *1* 卷：长篇论文），页码 450–482，墨西哥城，墨西哥，2024 年 6 月。计算语言学协会。doi：10.18653/v1/2024.naacl-long.26。URL https: //aclanthology.org/2024.naacl-long.26/。[20] Yuhan Liu、Michael JQ Zhang 和 Eunsol Choi。人类与大语言模型对话中的用户反馈：理解用户的视角，但作为学习信号存在噪声。载于 Christos Christodoulopoulos、Tanmoy Chakraborty、Carolyn Rose 和 Violet Peng 编，*2025* 年自然语言处理实证方法会议论文集，页码 2666–2681，中国苏州，2025 年 11 月。计算语言学协会。ISBN 979-8-89176-332-6。doi：10.18653/v1/2025.emnlp-main.133。URL https://aclanthology.org/2025.emnlp-main.133/。[21] Yu. A. Malkov 和 D. A. Yashunin。使用分层可导航小世界图进行高效且稳健的近似最近邻搜索，2018 年。URL https://arxiv.org/abs/1603.09320。[22] Gary Marchionini。探索性搜索：从查找到理解。计算机协会通讯，49(4)：41–46，2006 年。doi：10.1145/1121949.1121979。[23] Sunghyun Park、Han Li、Ameen Patel、Sidharth Mudgal、Sungjin Lee、Young-Bum Kim、Spyros Matsoukas 和 Ruhi Sarikaya。用于从隐式用户反馈中学习、以改进大规模对话式人工智能系统自然语言理解能力的可扩展框架。载于 Marie-Francine Moens、Xuanjing Huang、Lucia Specia 和 Scott Wen-tau Yih 编，*2021* 年自然语言处理实证方法会议论文集，页码 6054–6063，线上及多米尼加共和国蓬塔卡纳，2021 年 11 月。计算语言学协会。doi：10.18653/v1/2021.emnlp-main.489。URL https://aclanthology.org/2021.emnlp-main.489/。[24] Liana Patel、Siddharth Jha、Melissa Pan、Harshit Gupta、Parth Asawa、Carlos Guestrin 和 Matei Zaharia。语义算子：一种用于丰富的基于人工智能的数据处理的声明式模型。2024 年。URL https://api.semanticscholar.org/CorpusID: 271218837。[25] Mark Raasveldt 和 Hannes Mühleisen。DuckDB：可嵌入的分析型数据库。载于*2019* 年数据管理国际会议论文集，SIGMOD '19，美国纽约州纽约市，2019 年，第 1981–1984 页。计算机协会。ISBN 9781450356435。doi：10.1145/3299869.3320212。URL https://doi.org/10.1145/3299869.3320212。[26] Stephen Robertson 和 Hugo Zaragoza。概率相关性框架：BM25 及其拓展。信息检索前沿，3(4):333–389，2009年4月。ISSN 1554- 0669。doi: 10.1561/1500000019。URL https://doi.org/10.1561/1500000019。[27] Tefko Saracevic。重新审视相关性。载于 Peter Ingwersen 和 Niels Ole Pors 编，图书馆与信息科学观念国际会议第二届会议论文集（*CoLIS 2*），第201–218页，丹麦哥本哈根，1996年。皇家图书馆学学院。[28] Murray Shanahan、Kyle McDonell 和 Laria Reynolds。大型语言模型角色扮演。《自然》，623(7987):493–498，2023年11月。ISSN 1476-4687。doi: 10.1038/s41586-023-06647-8。URL https://doi.org/10.1038/s41586-023-06647-8。[29] John W. Tukey。探索性数据分析。Addison-Wesley，马萨诸塞州雷丁，1977年。ISBN 0-201-07616-0。[30] Matthias Urban、Jialin Ding、David Kernert、Kapil Vaidya 和 Tim Kraska。利用历史用户反馈提升文本转 SQL 的准确性。载于人在回路数据分析研讨会论文集，HILDA '25，美国纽约州纽约市，2025年。计算机协会。ISBN 9798400719592。doi: 10.1145/3736733.3736739。URL https://doi.org/10.1145/3736733.3736739。[31] Zilong Wang、Hao Zhang、Chun-Liang Li、Julian Martin Eisenschlos、Vincent Perot、Zifeng Wang、Lesly Miculicich、Yasuhisa Fujii、Jingbo Shang、Chen-Yu Lee 和 Tomas Pfister。Chain-of-Table：在推理链中演进表格以实现表格理解。载于第十二届国际学习表征会议，2024年。URL https://openreview.net/forum?id=4L0xnS4GQM。[32] Wes McKinney。用于 Python 统计计算的数据结构。载于 Stéfan van der Walt 和 Jarro d Millman 编，第九届 *Python* 科学会议论文集，第56–61页，2010年。doi: 10.25080/Majora-92bf1922-00a。[33] Ryen W. White 和 Resa A. Roth。探索性搜索：超越查询—响应范式。信息概念、检索与服务综合讲座。Morgan & Claypool 出版社，2009年。ISBN 978-1-59829-783-6。doi: 10.2200/ S00174ED1V01Y200901ICR003。[34] Ryen W. White 和 Resa A. Roth。探索性搜索：超越查询—响应范式。信息概念、检索与服务综合讲座，1 (1):1–98，2009年。doi: 10.2200/S00174ED1V01Y200901ICR003。[35] Shunyu Yao、Jeffrey Zhao、Dian Yu、Nan Du、Izhak Shafran、Karthik Narasimhan 和 Yuan Cao。ReAct：让语言模型中的推理与行动协同发挥作用。载于国际学习表征会议（*ICLR*），2023年。[36] Amy X. Zhang、Michael Muller 和 Dakuo Wang。数据科学工作者如何协作？角色、工作流程与工具。*ACM* 人机交互会议录，4 (CSCW1)，2020年5月。doi: 10.1145/3392826。URL https://doi.org/10.1145/3392826。[37] Shiqi Zhang、Xinbei Ma、Zouying Cao、Zhuosheng Zhang 和 Hai Zhao。Plan-over-graph：迈向可并行化的大语言模型智能体调度，2025年。URL https://arxiv.org/abs/2502.14563。[38] Xiaokang Zhang、Sijia Luo、Bohan Zhang、Zeyao Ma、Jing Zhang、Yang Li、Guanlin Li、Zijun Yao、Kangli Xu、Jinchang Zhou、Daniel Zhang-Li、Jifan Yu、Shu Zhao、Juanzi Li 和 Jie Tang。TableLLM：通过大语言模型实现表格数据操作，适用于真实办公场景。载于 Wanxiang Che、Joyce Nabende、Ekaterina Shutova 和 Mohammad Taher Pilehvar 编，计算语言学协会论文集：*ACL 2025*，第10315–10344页，奥地利维也纳，2025年7月。计算语言学协会。ISBN 979-8-89176-256-5。doi: 10.18653/v1/2025.findings-acl.538。URL https://aclanthology.org/2025.findings- acl.538/。[39] Yunjia Zhang、Jordan Henkel、Avrilia Floratou、Joyce Cahoon、Shaleen Deep 和 Jignesh M. Patel。ReAcTable：增强 ReAct 以实现表格问答。*VLDB* 学报，17(8):1981–1994，2024年4月。ISSN 2150-8097。doi: 10.14778/ 3659437.3659452。URL https://doi.org/10.14778/3659437.3659452。[40] Xuanhe Zhou、Zhaoyan Sun 和 Guoliang Li。DB-GPT：大语言模型遇见数据库。数据科学与工程，9(1):102–111，2024年3月。ISSN 2364-1541。doi: 10.1007/s41019-023-00235-6。URL https://doi.org/10.1007/s41019- 023-00235-6。[41] Dongsheng Zhu、Weixian Shi、Zhengliang Shi、Zhaochun Ren、Shuaiqiang Wang、Lingyong Yan 和 Dawei Yin。先分解后聚合：一种通过并行工具调用实现的高效工具学习方法。载于 Wanxiang Che、Joyce Nabende、Ekaterina Shutova 和 Mohammad Taher Pilehvar 编，计算语言学协会第*63*届年会论文集（第一卷：长篇论文），第28859–28875页，奥地利维也纳，2025年7月。计算语言学协会。ISBN 979-8-89176-251-0。doi: 10.18653/v1/2025.acl-long.1401。URL https://aclanthology.org/2025.acl-long.1401/。