snowflake®

建造
互操作湖屋

人工智能领导者数据策略

目录

人工智能时刻需要互联架构

当今，企业正面临人工智能的命令式，即要从使用人工智能中寻求可衡量的价值与效率的压力。他们正在超越实验阶段，进入生产阶段，最终发现成功的AI并非从模型开始——而是从数据开始。这正是为何以信心迎接这一时刻的关键恰恰在于您数据基础设施的准备好程度。

将数据准备为人工智能使用，然而，需要经过多个流程和考虑。这包括标准化元数据、跟踪所有数据源的可追溯性和质量检查，所有这些都在努力使数据始终清洁、精心整理、标记、可访问并在系统间进行管理。但在实践中，许多企业最终拥有跨越系统、云和地区碎片化和重复的数据资产，这使得其难以扩展，几乎无法进行管理。看到公司利用不同的数据仓库、数据湖和引擎并不罕见，因为每个团队或业务单元都是孤立地构建其架构，以在首选的堆栈上标准化。这种方法迫使中央数据团队在平台之间切换或创建薄弱的联系。

但有一条更好的方法，那就是结合两种优点的架构，它既能够根据每个数据客户的喜好提供灵活的配置，又能保持数据中心所有权、普遍的治理以及提供人工智能准备能力。请进入可互操作的数据湖屋。

湖泊屋建筑模式并非新概念，但传统方法缺乏让组织完全掌控其数据的根本机制。双向互操作性意味着能够安全地将任何引擎带入你的数据或访问任何数据以进行读写操作。可互操作的湖泊屋建立在开放标准之上，旨在实现供应商中立连接，企业可以自由构建而不妥协——在一个平台上实现无缝互操作性、加强治理并解锁企业级AI。

数据架构难题：企业为何停滞不前（或认为自身处于停滞状态）

随着组织规模的扩大，看到各个团队为满足自身特定需求建立流程并不罕见。营销和销售可能会转向数据仓库，而数据科学则优先考虑数据湖。再加上企业级SaaS工具的庞大生态系统，如Workday和Salesforce，它们都在产生对商业AI项目至关重要的数据。然而，结果却是，现代技术栈与其说是堆叠，不如说是迷宫。每个新工具都增加了整体系统的复杂性，尽管它可能必要地在一个区域上减少了摩擦。但随着每一次技术升级和在每个新层上增加，工程团队很快就会意识到他们陷入了一个由碎片化系统构成的复杂迷宫中。

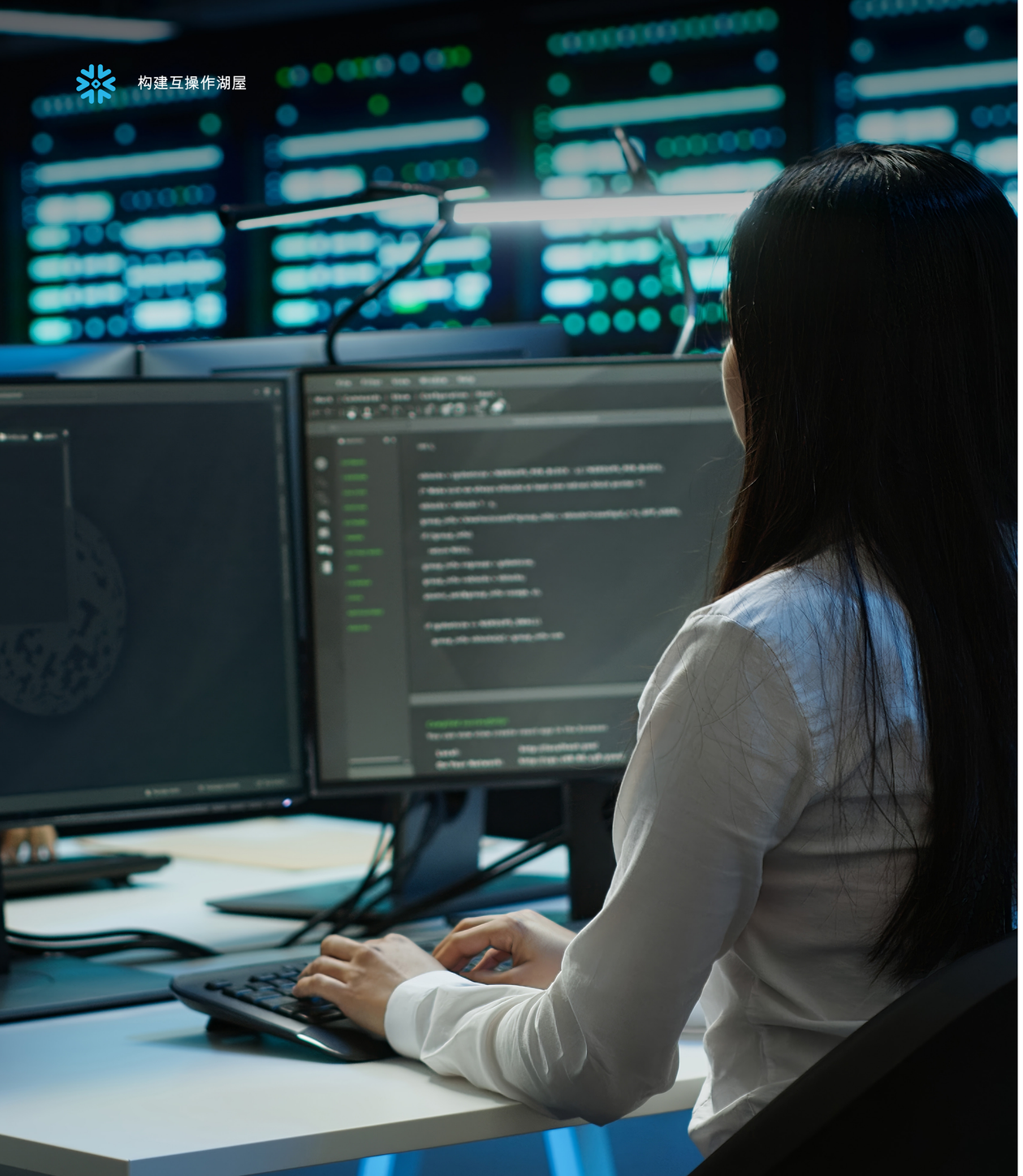
这意味着当一个系统出现故障时，他们会打开一道又一道门，只是寻找通向问题的正确途径。在某些情况下，他们只是最终建立了一条新路径，这是一种短期解决方案，通常会导致脆弱、昂贵且耗时的管道，这些管道依赖于数据复制。这不仅会引起对真相来源的无谓困惑，还限制了人工智能的效果和准确性。

本质上，这些系统创建了一个反应式维护循环，迫使数据工程师充当成语中的消防员，仅仅希望在全面火灾爆发前跟上每一场新火势。而且随着复杂架构在多个引擎和多个云中蔓延，这些系统可以造成巨大的运营阻力。

这通常导致建筑师在需要大量运营开销的灵活DIY基础设施和强制一定程度的供应商锁定管理的解决方案之间做出选择。任何一种选择都伴随着重大的权衡——最显著的是可靠性和成本。

但是，这里有个问题：这是一个错误的选择。

实际上，缺失的遗产湖屋方法实际上是可以实现的：一个连接但开放的数据和治理基础，为团队提供选择他们喜欢的引擎和工具的灵活性，同时集中治理和语义。



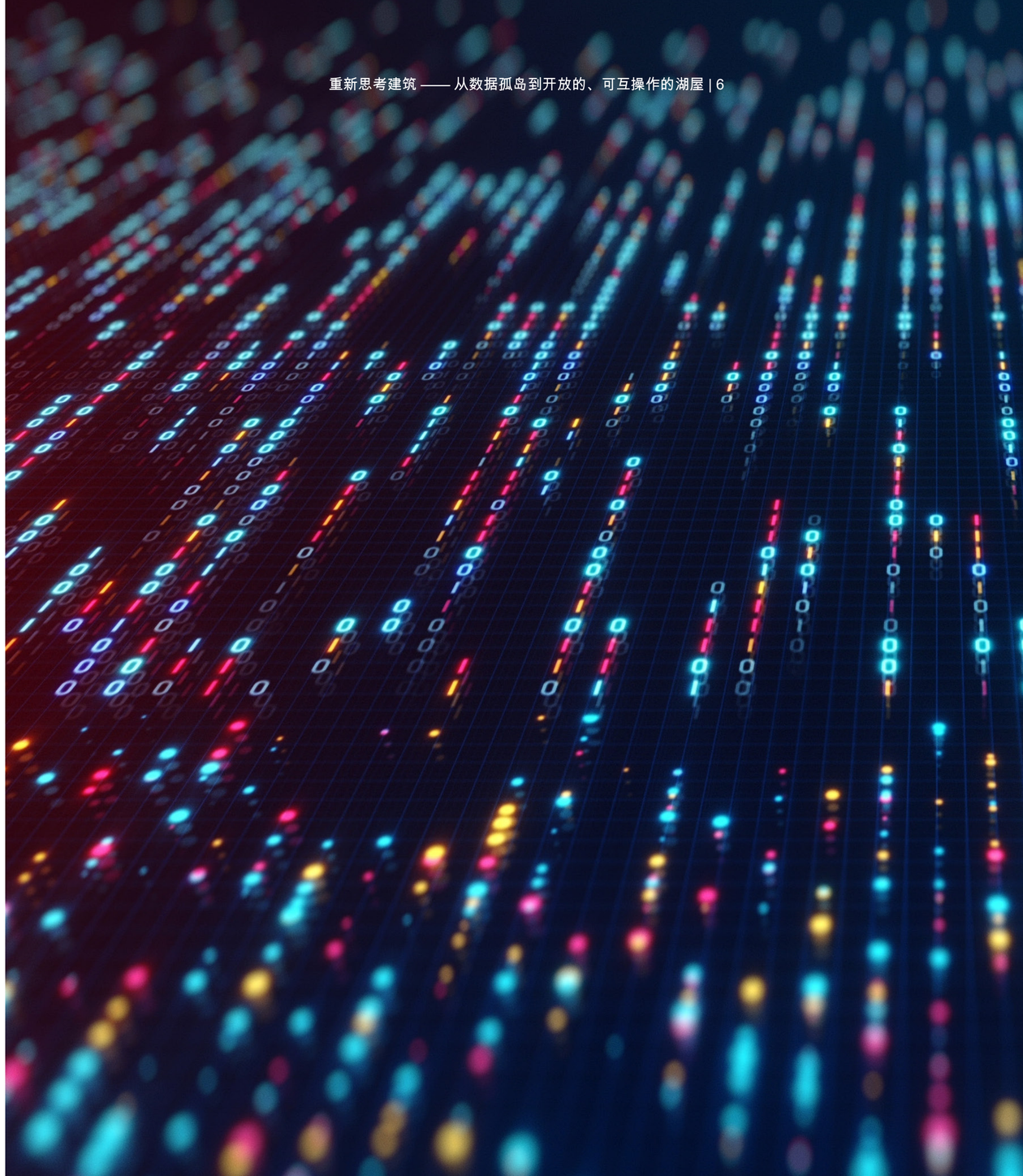
在这本书中，我们将探讨开放且可互操作的数据湖屋——作为一个概念，其起源以及一些旨在帮助企业成功利用人工智能而非仅仅对其做出反应的最佳实践。我们将探讨支撑这一架构的三个基本支柱——双向互操作性、规模化简化和人工智能的通用治理——所有这些都是为了展示专为使任何企业人工智能就绪而设计的湖屋几乎无限的威力。我们将分享真实企业如何通过在Snowflake上建立正确的湖屋基础来实现人工智能投资回报率的故事，并从我们的主要云合作伙伴亚马逊网络服务（AWS）、谷歌云和微软那里提供见解和架构模式。

这本实用指南旨在向你们展示企业数据中心湖如何帮助让人工智能的实现更加头痛减少、成本更低。我们将展示企业如何重新获得对本数据中心资产的完全所有控制权，摆脱那种所谓的迷路的恐惧。

重新思考建筑——从数据孤岛到开放的、互操作的湖屋

许多企业今天都在过多信息的问题上挣扎。整理这一切很容易感觉像是一场西西弗斯的劳作，无穷无尽地管理着结构化数据的数据库，同时还有非结构化图像、视频或文档的数据湖。在实践中，这往往导致更多的隔阂，迫使架构师陷入复杂性。然而，随后湖屋（lake house）的出现有助于解决这些问题——带来了性能、灵活性、成本效益以及降低供应商锁定风险的前景。

那么，什么是可互操作的数据湖屋呢？



数据仓库 vs. 液态屋 vs. 数据湖

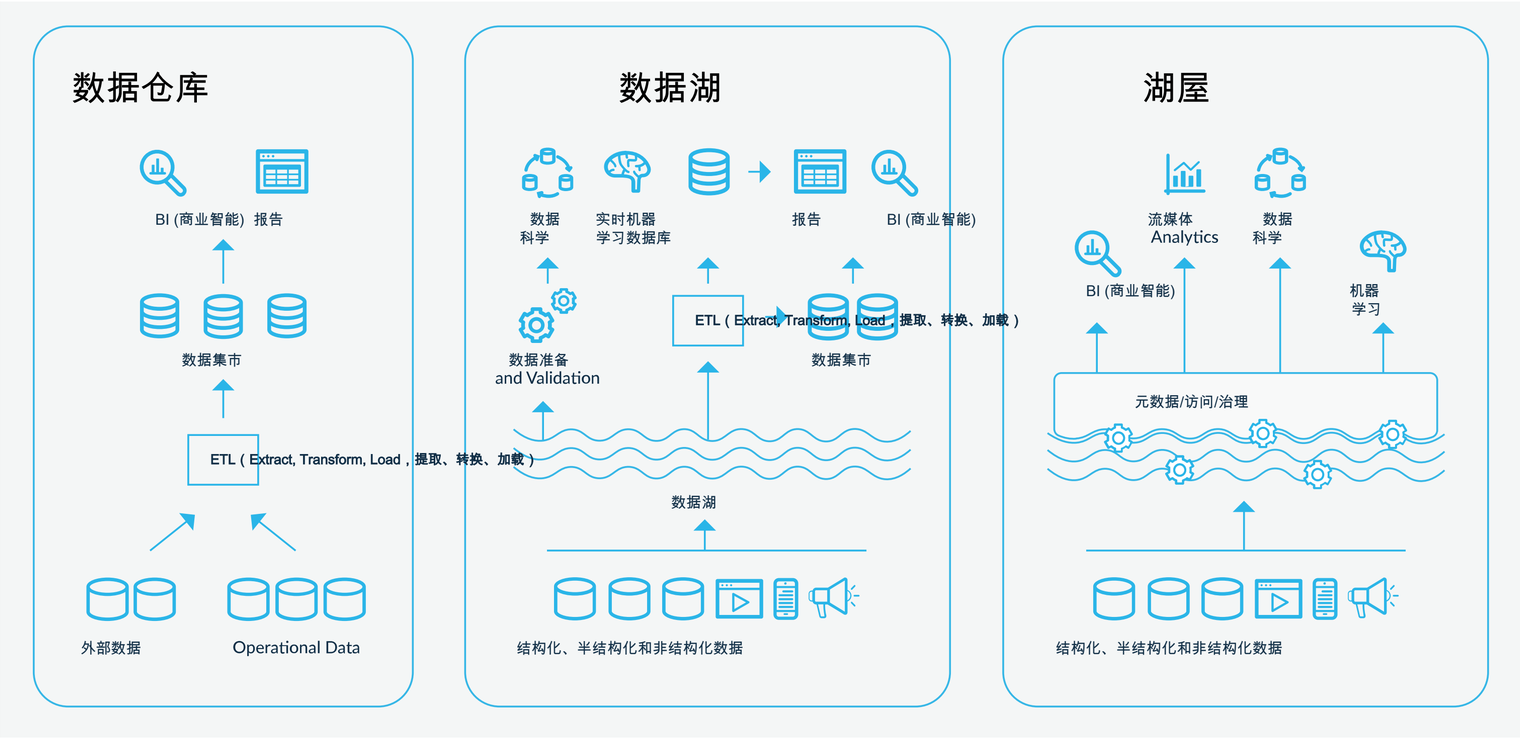
为了帮助您理解传统湖屋和互操作湖屋之间的区别，我们首先需要回顾一下当今大多数数据系统中存在的三种核心架构。想想人们是如何使用他们的车库来储存物品的。想想那些极度有条理的人，他们沿着每面墙都安装了内置的架子，并把他们所有的物品整齐地收纳进每个都清晰标记的塑料箱中。如果有什么东西放不进箱子里或抽屉里，就会被丢弃。这就是数据仓库，一个用于存储结构化和有序数据的存储库。

现在，想象一下一个收藏狂，他们把车库当成了一切物品的垃圾场：DVD堆在旧纳税申报单的箱子上；一辆自行车侧躺，周围散落着各种体育纪念品，积满了灰尘。

这种能够容纳各种结构数据存储方式被称为数据湖。

然后是第三种类型：数据湖屋。继续这个比喻，这个车库沿墙有可调节的金属架子，还有几个井然有序的塑料箱子，但并不是所有东西都能整齐地放进盒子里——这没关系。总的来说，车库是井井有条的，每个物品都有自己的位置。

正如人们所想象的那样，每种架构都有其自身的优点（即使是数据湖——它具有灵活性和愿意容纳各种东西的能力！），因此评估哪种架构适合任何特定情况变得非常重要。但对于许多具有前瞻性的组织来说，一个越来越明确的事实是，互操作性的湖屋正在证明自己是 选项实现了灵活性的承诺 *现代化*
并且扩大规模，同时不牺牲可靠性和性能。



比较架构

	数据仓库	数据湖	数据湖屋
数据类型	结构化	结构化 半结构化 非结构化的	结构化 半结构化 非结构化的
表现	High	可变/低	High
治理	强大/封闭	困难/手动	强烈/开放

表1. 快速了解为何湖屋成为AI驱动企业的标准。

一个成功的湖屋被定义为三个核心特征：（1）开放式表格格式（例如Apache Iceberg、Delta或Hudi）的使用；（2）跨引擎互操作性；（3）集中式治理结构。

什么是使湖边住宅互操作的关键？

太频繁的，湖景房子的建筑因设计决策基于不同的开放式表格格式而无法全面实现数据所有权和互操作性。

开式表格格式（在第 following 章节中将有更深入的解释）是关于从兼容的计算引擎访问磁盘上的文件应该如何行为的规范。开式的分类来自于满足开源软件的基本要求。两种显着的开式表格格式为定义你的 Lakehouse 对大多数现代架构的开放性和互操作性提供了基准：Delta Lake 和 Apache Iceberg™。

Delta Lake方法

Delta Lake 是一个针对 Apache Spark™ 工作负载优化的开源开放表格格式。因此，典型的实现提供了一种单写多读的模型。这意味着这个表格非常适合寻求在单一供应商或作者上标准化的企业，但在尝试实现开放湖仓和表格格式所承诺的完全双向互操作性时，会遇到摩擦。

结果是，尽管Delta Lake的湖屋建立在开源技术之上，但这种开放性是有限的。这些遗留的湖屋最终还是将用户束缚在单一供应商手中。这个领域的供应商通常要求特定的目录来实现完全功能，或者限制写入能力，从而阻止其他工具更新或管理数据。同样，他们将关键的性能优化仅限于自己的专有计算引擎。

不久之后，人们发现表面上看似开放的架构在许多重要方面仍然被封闭。原本旨在被打破的数据孤岛依然存在，只是形式不同，结果是功能锁定，这最终迫使进行迁移和数据移动。突然之间，将工具带到数据而不是将数据带到工具的好处几乎丧失殆尽。

Apache Iceberg™方法

这正是Apache Iceberg——凭借其供应商中立、互操作性——成为一大启示的原因。与Delta Lake不同，Iceberg提供多写多读的规范。这意味着客户可以在一个平台上实现标准化，同时利用他们偏好的计算引擎和工具进行任何操作。

要了解更多关于冰山的信息，请参阅第11页的边栏。

事实是，如今仅仅声称开放是不够的；企业湖屋必须具备开放性和互操作性。 *并且*

雪花的Lakehouse企业愿景

一个建立在双向互操作性基础上的开放和兼容的湖屋，就像是没有Wi-Fi接入或SIM卡的高端智能手机一样；没有连接的情况下，它的实用性会大大受限。这正是为什么Snowflake不断地考虑系统如何协作——以及如何做到更好。

这始于对Iceberg表的通用本地读写支持——无论目录、区域还是云。现在，无论表在哪里，用户都可以在完整性能下进行数据交易、查询和修改，无需其他平台通常要求的只读限制或复杂的摄入管道。

为了使这种架构真正达到企业级水平，Snowflake在该开源基础上叠加了多项关键功能：

- 地平线目录 带有Apache Polaris™核心，用于治理：这提供了一个连接您所有数据并与所有引擎兼容的治理解决方案，让您能够构建一个开放的、基于元数据的治理和安全模型，无需锁定即可在表格、行和列级别隔离数据。

- 目录关联数据库： 联邦远程IRC兼容目录，实现自动发现和数据刷新，构建统一的受控视图，并从Snowflake对任何Iceberg数据进行操作，无论目录或存储位置如何。

- Cortex Code CLI 和 雪花智能 赋予您的技术和业务团队使用Snowflake原生智能和全面管理的平台，将复杂的数据、互操作性和基础设施任务提炼成简单的自然语言交互和自动化的能力。

- 雪花存储用于冰山表：将用于本地表的零管理存储模型扩展到冰山格式，解锁所有互操作性，而无需管理云存储的复杂性。

- 皮质AI用于丰富化：用户可以将先进的人工智能和机器学习模型直接带到数据所在之处，从开放表中直接提取洞察，无需数据迁移。

- 滑雪公园 为了计算灵活性：这允许数据工程师和科学家在Snowflake的安全范围内使用他们偏好的语言（例如Python、Java或Scala）处理开放数据。

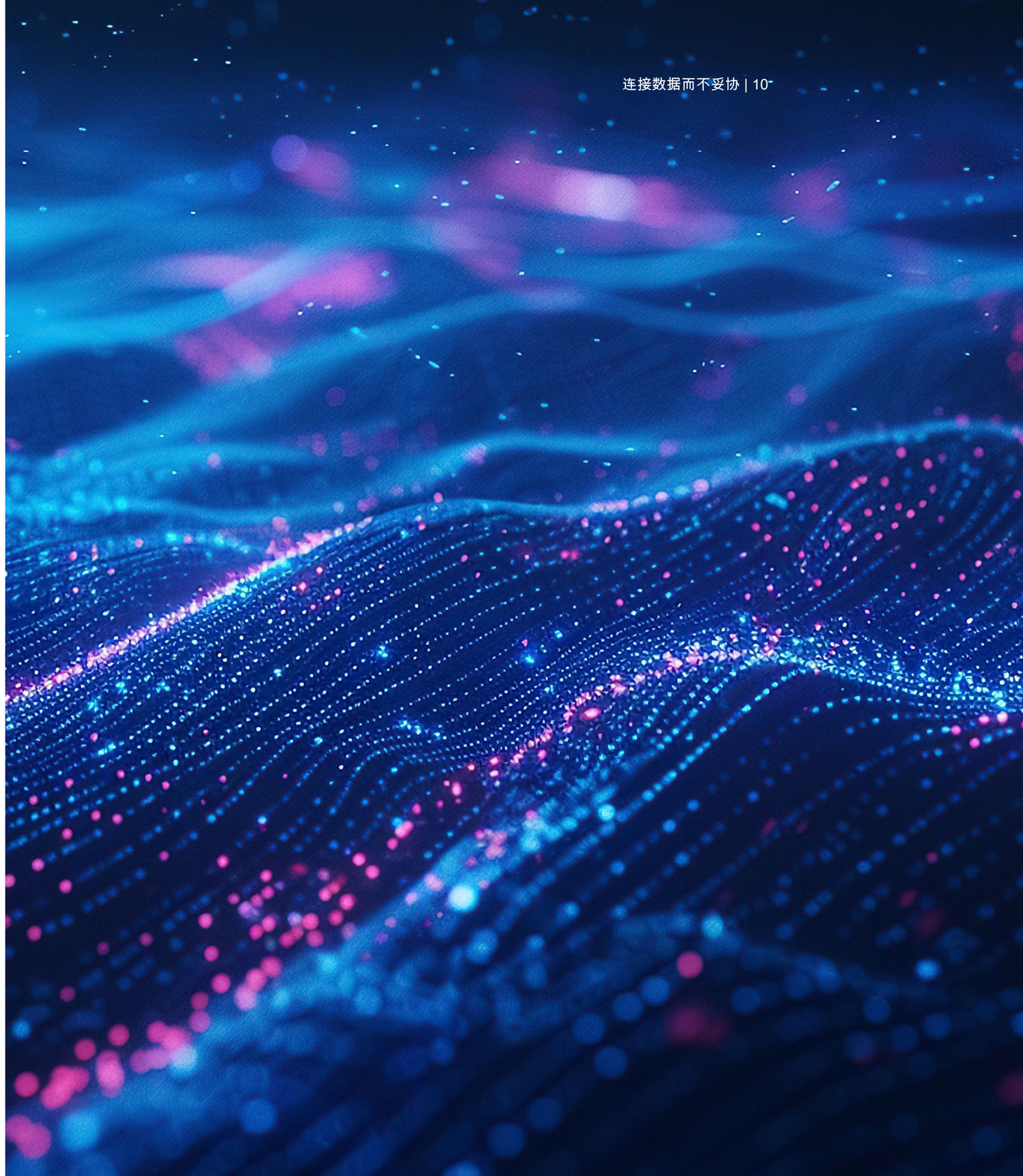
- 企业业务连续性 Snowflake将其跨云复制和故障转移功能扩展到Iceberg表，确保开放标准不会以业务连续性或灾难恢复为代价。

这所有的目的在于建立一个具有互操作性的数据湖，需要更少的组件，易于管理和控制，并真正避免锁定。这是将数据所有权归还给企业——它本应归属的地方，并最终实现开放数据湖一直承诺的好处。

无妥协连接数据

曾经那谦虚的数据库稳固地矗立在此，企业如今已建起宏伟的数据王国。面对来自无数不同来源的众多不同数据类型，能够使其全部标准化并可供各种系统和目的使用，归根结底就是——没错——互操作性。

这就是这种湖屋的基本美：它通过数据API、Apache Iceberg规范支持多个云和许多引擎，作为唯一的连接组织。然而，挑战在于，如今的数据领地是碎片化的；它们的连接点薄弱。实际上，每个新的业务数据请求都变成了一项复杂的项目。数据团队最终成为进步的瓶颈，加深了它阻碍创新的印象。因此，为了解决这个问题，我们必须从存储开始。



开放表格格式简介

为了理解开放表格格式的重要性，从数据湖的不足之处开始是有帮助的。如前所述，数据湖有许多优点——其中包括成本效益、灵活性和可扩展性——但也有显著的权衡。因为它们本质上是一个没有共同事务元数据层的文件集合，数据湖无法原生支持ACID（原子性、一致性、隔离性、持久性）事务。这意味着，如果在写入过程中工作失败，它就会戛然而止；然后，你将面临半成品、损坏的数据。而且因为没有版本控制，清理这些数据需要花费大量时间和精力。此外，数据湖的“读取时模式”方法常常导致管道故障，因为即使是一点小的模式更改——添加一个列或更改数据类型——也会导致下游所有内容失败。这些问题带来的麻烦比解决方案多，不久之后，许多数据湖变成了浑浊、混乱的数据沼泽。

开放式表格式，然而，再次出现以帮助澄清谚语般的浑浊水域。这些现代格式，包括Apache Iceberg、Delta Lake和Apache Hudi，提供了一层元数据，使得ACID事务、模式演化和时间旅行成为可能。它们还支持多引擎访问，以便不同的查询和处理引擎可以针对一致的表表示进行工作。但如前所述，选择正确的开放式表格式对于互操作性至关重要。Apache Iceberg（见侧边栏），特别是由于其支持任何操作的全套供应商中立访问，除ACID保证和高效的文件级操作外，已经获得了巨大的吸引力。这允许分析引擎在规模上自信地读取和写入共享数据集，将原始文件集合转换为一个可靠、可治理和互操作的数据层。

为什么冰山？

Apache Iceberg在众多开源表格格式中脱颖而出，因为它从第一天起就是围绕互操作性、开放性和引擎独立性进行设计的——不受任何单一计算引擎或供应商路线图的限制。

在本质上，Iceberg 是以规范为先。也就是说，它是通过一个开放且严格管理的规范来定义的，而不是通过参考实现。这意味着任何引擎——无论是 Spark、Flink、Trino、Snowflake、Hive 还是未来的引擎——都可以一致地实现 Iceberg，无需逆向工程。相比之下，Delta Lake 和 Hudi 历来与其主要引擎紧密耦合，这可能会微妙地影响有效互操作性。

冰山的元数据模型是另一个关键差异化点。隐藏分区、原子快照、时间旅行和行级删除等功能以干净、与引擎无关的方式实现，避免了查询端的复杂性以及分区泄漏。模式和分区演变是一等、可预测且安全的。

兼容性承诺通过Iceberg的目录抽象得到进一步加强，包括对Iceberg REST目录规范的支持，实现了跨云、引擎和治理系统的表访问一致性。这使得Iceberg成为开放式数据架构和多引擎分析的理想基础。

简而言之，冰山不是 *仅仅* 表格格式——这是数据与计算之间的一种耐用的合同。这种中立性，加上成熟的功能和广泛生态系统采用，使得Iceberg成为最强长期选择。

这些表格，然而，需要一个目录作为元数据存储库和所有数据湖中数据的权威真相来源。目录不是将所有表元数据内部存储，而是维护每个表的当前元数据文件的指针，并对其进行原子更新（使用比较和交换操作）。当涉及到冰山目录——例如，Apache Polaris™——这个文件包含了完整的快照历史、模式、分区规范和文件清单，描述了实际数据文件存储的位置。这并不完全不同于图书馆卡片目录，或者说是您数据的一个GPS系统。

为了实现广泛的互操作性，目录应实施Iceberg REST目录规范，这是一个标准API，它允许任何兼容的引擎（如Snowflake、Trino、Spark或Flink）与您的Iceberg表进行一致交互。选择遵循此规范的目录对于保持数据的单一、受控副本至关重要。

开放式表格格式的巨大优势

最终，使用这种架构消除了碎片化并减少了数据迁移的需求。这意味着通过就地访问数据来更快地上线，同时降低与数据移动相关的成本。能够对单个数据副本进行工作，无论其位置如何，都将减轻工程团队的工作负担，使他们能够专注于创新，而不是构建和维护管道。

三项构建Apache Iceberg的最佳实践

构建高性能互操作湖仓，使用Iceberg不仅需要摄取数据，还需要对您的表布局和元数据进行主动管理。

1. 使用隐藏分区：与Hive不同，您必须手动为分区创建额外列（例如：*处理日期* 源自 *时间戳*），冰山（Iceberg）原生支持此功能。使用隐藏分区来定义分区转换，例如 *day(timestamp)*，直接在表定义中进行。这使得任何引擎都可以自然地查询源列，无需知道物理分区方案。同时，引擎会自动修剪不必要的文件。

2. 解决“小文件”问题：流式摄取通常会产生成千上万的微小文件，这会降低读取性能并使元数据膨胀。实施常规压缩策略（使用
重新编写数据文件 在Spark或 *优化* 在Trino中将小文件合并成更大、读取优化的文件。针对128 MB至1 GB的文件大小用于分析工作负载，以平衡并行处理与开销。

3. 定期过期快照：冰山的时间旅行功能强大，但无限期保留每个历史快照会导致元数据膨胀并减慢查询规划。设定保留策略，以过期那些超过所需审计窗口（例如，7天或30天）的快照，并删除孤立文件。这可以使您的元数据保持轻量级，并控制存储成本。

简化规模化的声明式数据处理

有效地整合和组织数据至关重要，但让数据为你工作才是真正目的所在。最终，仅仅存放在存储中的数据是没有价值的；它需要成为工作流程和应用程序的一部分。它需要被分析和应用。只有这样，组织才能从……
拥有 数据实际上 *受益* 从它。

但是，在数据湖中，来自众多来源的格式多样的数据可能不断流入，构建、扩展和维护管道变得复杂。工作流程开始像迷宫一样，复杂性导致崩溃和失败。数据工程师发现自己始终处于防守状态，修补而非现代化。他们不断管理大型实时管道的基础设施，并不断调整以提升速度、可靠性和成本。这种显著的摩擦不可避免地耗尽了宝贵资源，并束缚了数据工程团队的发展。

从基础设施难题到ZeroOps便捷

借助生成式AI和其他自动化技术，零运维基础设施管理的梦想比您想象的更接近现实。一些重要的发展——例如AI智能体、声明式管道和零复制数据——帮助提高了效率，同时不牺牲安全性和可靠性。它们允许您通过自动管理您基础设施的工具来扩展您的管道（而不会增加您的成本）。

Cortex代码命令行界面

为了帮助开发者更快更准确地编写代码，Snowflake推出了Cortex Code，这是一款Snowflake原生的人工智能编码代理，旨在将复杂的互操作性、数据工程、分析、机器学习和代理构建任务转化为简单且信息丰富的交互，以实现高准确性和信任——所有这一切都通过自然语言完成。

尽管第三方通用编码代理功能强大，但它们缺乏提供上下文感知自动化工作流所需的Snowflake元数据、目录信息和基于角色的访问控制的原生意识。相比之下，Cortex Code专为加速整个数据生命周期，并具备企业级治理和可靠性而构建。

这一深度平台智能通过加速数据工程、高级分析和代理及应用的开发，缩短了产品上市时间。它还赋予所有用户——从技术专家到非技术人员——基于数据的自信建设能力。

复杂任务和复杂的流程被简化，有助于全面提升生产效率。

声明性管道

传统上，数据工程师花费数小时耐心编写数据管道的操作指南。然而，采用声明性方法后，数据编排的定义是由实现的结果来决定，而不是由 它实现了（例如，一个） *怎么样*
指令式管道方法）。如果你想到指示方向，声明式方法会是：“去杂货店。”指令式方法则是：“沿主街向东行驶；在榆树街左转；继续行驶三个街区；右转进入杂货店停车场。”

Snowflake's 动态表格 提供这样的声明式方法。您不需要定义有向无环图（DAG）的每一步，只需定义您数据期望的最终状态。雪崩的查询优化器随后会自动确定刷新数据的最有效方式，并在幕后管理调度和依赖关系。这减少了管道维护的手动负担，并确保数据新鲜度，而无需额外的开销。雪崩的Iceberg动态表在读取操作上保持与任何引擎的互操作性，但需要一个单独的写入者以避免冲突、损坏和重复更新。

自动化流式摄取和转换

现代湖上住宅需要数据一到达就准备好分析。例如，像这样的工具： 雪管 并且 雪管流
支持Apache Kafka的架构演化，实现数据到达的“设置即可忘记”的便捷方式。Snowflake

自动检测传入的JSON或Parquet文件中的模式变化，并相应地调整目标表。除了使用动态表进行声明性转换外，数据工程师还可以通过无服务器任务自动化转换层。组织可以通过自动编排和基础设施管理，从批量处理转移到连续、近乎实时的洞察。

此外，Snowflake与云服务提供商合作，集成如Amazon Data Firehose、Amazon MSK和Azure Event Hubs等流产品，为联合客户提供更快的洞察时间。

雪花还通过Snowpark Connect for Apache Spark添加了对Apache Spark工作负载的支持，这使得用户能够享受到雪花向量化引擎的性能和管理服务的简便性。这种方法解决了管理旧版Spark管道的核心挑战——集群维护的高运营成本、安全治理的碎片化和不可预测的出口成本——同时保留你的代码库和技能。你可以无缝地将现有的Spark工作负载迁移到雪花，无需进行最小或无代码更改，几乎消除了配置、调优或维护单独Spark计算集群的需求。雪花的引擎处理所有这些，包括优化查询执行而无需用户干预。这减少了调优开销，可以带来显著的成本节约。 一些客户报告节省了41%的费用 .

零拷贝数据

在最大的“基础设施难题”中，与开发、测试或共享数据相关联的成本和风险是最显著的。Snowflake的无副本、双向合作伙伴集成基本消除了这些问题。通过多集群共享数据架构，Snowflake允许您创建生产环境的即时副本——无论规模大小——而无需实际复制底层存储。

这允许：

- 更快的创新：开发者可以在无风险的情况下测试生产级数据。
- 降低成本：您只需支付存储费用： *修改后* 块，而不是整个副本。
- 无缝协作：通过Snowflake Horizon与合作伙伴共享数据是通过指针访问实现的，确保了安全性和“单一真相来源”的完整性，无需进行任何ETL作业。

零拷贝、双向集成：数据与行动的桥梁

即便有了可互操作的数据湖基础架构，许多组织仍然难以将数据转化为行动。挑战在于在不依赖脆弱的管道或重复数据的情况下保持关键系统的统一。当运营和分析平台失去同步时，团队会对他们的见解失去信心，并减缓其响应能力。

Snowflake与SAP、Salesforce和Workday等战略合作伙伴提供零拷贝、双向集成，通过设计来填补这一空白。Snowflake与领先的 Enterprise Technology Providers紧密合作，建立共享的、受管理的分层数据层，在这里数据可以跨平台访问和共享，无需移动或复制。通过一个SQL对象、目录链接数据库和语义视图，每个记录系统都通过统一的平台连接到Snowflake。这使客户能够将他们最重要的运营数据连接到由Snowflake AI Data Cloud管理的平台，同时保持当前性、可信度，并随时可用。

变化持续流入您的Snowflake管理Iceberg表，用于分析和AI，同时洞察可以反馈到运营系统中，以指导下游流程。由于这种集成是双向的，数据在生态系统间保持一致，无需脆弱的手动交接或手动对齐。对于客户来说，结果是更敏捷的数据架构。数据团队花费更少的时间维护管道，更多的时间交付价值。通过中央管理的零副本集成，组织可以保持强大的安全性、语义智能和合规性，同时为人类和机器提供及时高效的数据访问。

通过无缝集成关键系统直接进入Snowflake，我们使湖屋不仅在理论上具有互操作性，而且在实践中可操作，将洞察力直接连接到行动。

一个强大的开发环境以满足多样化的需求。

开发者掌握着启用智能代理型应用程序的钥匙，但如今他们的俗称钥匙圈正变得拥挤不堪。如此多的任务和整合需求，他们不得不面对众多开发环境和工具的混乱，导致生产力下降和创新减慢。

为了高效构建现代智能应用程序，开发者需要一套快速、协作且易于使用的集成解决方案。随着 工作空间 ，在Snowsight UI的强大、统一编辑环境中，开发者获得了一个结合结构化代码组织、内置Git集成、Cortex Code、交互式图表等功能的一体化界面。用户可以选择在SQL或Python中进行工作，同时管理包括各种项目类型在内的项目。 dbt项目在Snowflake上．新进化出来的 雪花 CLI 提供广泛的命令行界面，可直接与Snowflake对象交互或构建可手动执行或按计划运行的自动化任务。具有使用任何工具的灵活性以及能够以编程方式控制Snowflake内部任何对象的权限，开发者可以轻松地在Snowflake环境中构建。

治理安全与信任以适应人工智能时代

管理分布式数据景观中的治理和安全是一个主要的运营难题。数据工程师陷入拼接多个目录、复制政策和维护持续的治理，这最终将他们的注意力从创新上转移开。但是，治理不应该是数据生命周期末尾的一个独立功能；它需要融入架构中。

传统的基于孤立的目录和针对特定系统的策略的途径，根本无法跟上步伐。每一款新引擎或数据集都带来另一个配置错误的机会；每一项政策的重复都导致偏移。治理最终变成一个永无止境的维护负担——随着数据资产的增长而加剧。

普遍的治理方法

为了充分利用湖景房的好处，统一的治理立场有助于建立对数据和管道的信任。Snowflake的Horizon目录就是为此设计的。它不是将治理视为辅助责任，而是将其定位为整个数据领地的连接组织。Horizon充当一个通用目录——一个元数据、政策、安全控制和世系会聚为一个单一、连贯系统的场所，该系统跨越云、区域和开放格式。无论数据存储在Iceberg、Delta Lake还是原生的Snowflake表中，治理模型都建立在这些数据之上，确保一致性，而不造成运营扩张。

这个转型的基础部分是朝着细粒度、自动化控制的转变。从历史上看，访问管理一直是采用补丁式的做法：一个工具中采用基于角色的访问，另一个工具中采用对象级权限，以及脆弱的脚本来在不同的环境中执行策略。Horizon用统一的模型来替代这种做法，该模型将基于角色的访问控制（RBAC）与属性驱动的精确度相结合。 那在所有引擎中都持续存在的 这提供了所需的隔离，适当时的灵活性，最重要的是，保持权限与访问控制一致性的信心。

自动化也改变了合规和数据质量的特点。不是依赖定期的审计或手工检查，Snowflake直接将治理机制融入运行架构。质量预期变成可定义的对象。当数据出现偏差时，预警会主动触发。敏感属性会自动被删除或屏蔽。合规状态不再是分散的日志和仪表板，而通过单一视角可见。这种效果是将从被动清理转变为主动保护——一种能够跟上现代数据管道和AI速度的治理，而不是滞后于它们。

人工智能治理

人工智能系统需要上下文才能繁荣发展。但当今的AI在元数据稀薄、不一致或缺失时往往难以应对。没有共享的定义、清晰的血统或有意义的文档，即使是复杂的代理也能误解数据所代表的内容，产生幻觉。为了产生可信赖的人工智能，企业必须帮助AI理解他们的数据，而不仅仅是访问它们。

Horizon 还提供了这一解释层。通过 语义视图自动驾驶 它将物理数据结构抽象为清晰的企业指标和实体。通过整合血缘关系和这些语义定义，丰富元数据并自动生成数据字典，它为人类和AI提供了可靠且高效的关于数据含义及其使用方法的了解。最终，这有助于减少幻觉，并鼓励AI系统负责任地行事，同时降低每条思考的成本。

在统一治理的基础上，数据交付变得显著简化。企业用户可以使用Snowflake Intelligence“与他们的数据交流”，减少通常会使数据团队陷入困境的临时性请求流。应用程序和BI工具通过安全连接连接。 语义视图 保留生态系统中跨业务环境的一致性，而不是依赖定制配置。企业可以将受控的数据产品——无论是基于Iceberg还是本地Snowflake表格——交付给任何利益相关者或外部合作伙伴，而无需移动或复制数据本身。

最后，治理才是将湖景屋转变为人工智能 ready 的基石。它向一系列数据集注入信任，让企业能够在不牺牲控制权的情况下进行创新扩展。

湖屋分析：Snowflake如何为周一早晨注入信心

星期一早上8点：仪表盘刷新；高管们登录；AI模型处理生产数据。这时候才能真正考验你的湖屋战略。

开放式架构提供了基础——支持互操作性、灵活存储和多引擎访问——但随着企业工作负载的增加，许多分析引擎开始感到压力。随着并发性的提升和工作负载对资源的竞争，性能变得不一致。

湖屋本身不是问题；问题是引擎。将Snowflake的引擎直接引入湖屋数据可以填补这一差距。以下是它如何缓解压力的：

消除摄入延迟：许多湖屋引擎需要数据迁移或额外的管道来实现可接受的速度。Snowflake查询直接打开Iceberg、Delta Lake和Parquet等开放格式，以实现无需数据迁移或重复的数据分析和高性能AI。

• 解决并发瓶颈：共享集群迫使BI、AI和运营工作量竞争相同的资源。Snowflake的独立虚拟仓库允许每个工作量独立扩展，减少竞争并保护服务等级协议，随着使用量的增加而保持其性能。

减少了持续的性能调整：随着数据量的增加，一些引擎需要持续手动优化以维持可靠性。Snowflake将性能优化内置到引擎中，自动处理元数据和并行执行，以减少运营开销。

• 在不分裂的情况下扩展治理：安全通常在不同目录和引擎之间被隔离。使用Horizon目录，治理策略在Snowflake管理的和外部管理的Iceberg表上均得到一致应用——加强控制而不限制互操作性。

• 通过AI实现自助式扩展：通过直接将所有数据连接到Snowflake而无需摄取，您可以赋予您的数据客户无论数据存储在哪里都能直接与之交流的能力，同时保持现有的治理和访问控制。

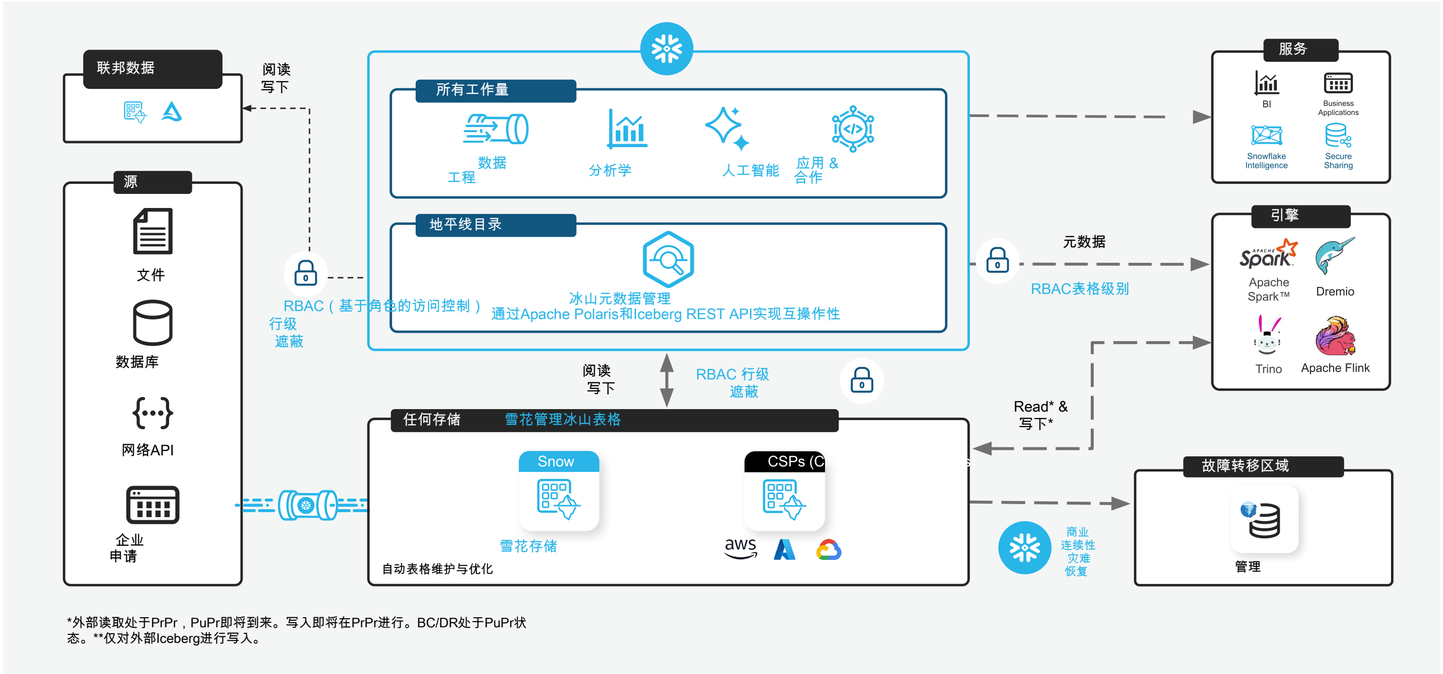
开放架构为您提供选择。将Snowflake的性能带入您的湖屋，确保这些选择在企业压力下仍然经得起考验，当性能、规模和信任最为关键时。

建筑领域的互操作湖仓架构

现在我们已经确立了强大互操作湖仓的三根支柱（双向互操作性、规模简化和针对人工智能的通用治理），让我们具体描绘一下这究竟是什么样子。根据您的技术栈、云服务提供商以及团队的需求，湖仓可以呈现出多种形态。

这里，我们展示了四个现代湖屋建筑的例子，突出了Snowflake与我们的主要云合作伙伴的互操作性，并从我们生态系统的各个领域的执行者那里带来了有见地的观点。

兼容的Snowflake湖泊屋



*外部读取处于PrPr，PuPr即将到来。写入即将在PrPr进行。BC/DR处于PuPr状态。**仅对外部Iceberg进行写入。

图1：雪花动力互操作湖屋图示。其核心架构组件包括：

1. 数据来源和摄入：数据从不同的来源摄入——包括文件、数据库、Web API和企业应用——进入Snowflake管理的Iceberg表中。

2. 任何存储（勋章式架构）：该架构遵循勋章图案（青铜、银色、金色）托管在云端存储上。

a. 青铜：未经处理的原始历史数据
b. 白银：经过整理的数据，Apache Iceberg在此促进大量转换和跨引擎的读写操作
c. 黄金：高性能、已处理、可供消费的数据

4. 雪花边缘目录：作为统一的治理和发现层。它通过Apache Polaris和Iceberg REST提供冰山元数据管理和互操作性。APIs。这使得外部引擎如Apache Spark、Dremio、Trino和Apache Flink能够以一致的控制访问权限访问数据。

5. 所有工作负载：Snowflake作为数据工程、分析、人工智能和协作应用程序的核心引擎。

6. 服务层：该架构支持多种下游消费者，包括BI工具（Power BI、Tableau）、商业应用和Snowflake Intelligence。为确保数据共享安全。

7. 业务连续性：包括集成故障转移区域，以实现灾难恢复和管理的连续性。

人工智能适应性是衡量数据资产健康状况的新指标。碎片化、孤岛化的数据现在构成了一个生存威胁，增加了您最重要的人工智能项目的Token浪费和幻觉风险。Snowflake的互操作性数据湖策略赋予您构建一个连通的数据基础，使得每个AI代理和人类驱动的流程都可以在实时受管数据上安全地、大规模地运行，无论其位于何地，都不会受到锁定。

——克里斯托弗·查尔德
副总裁，产品，雪flake

冰川湖畔雪晶之家与微软OneLake

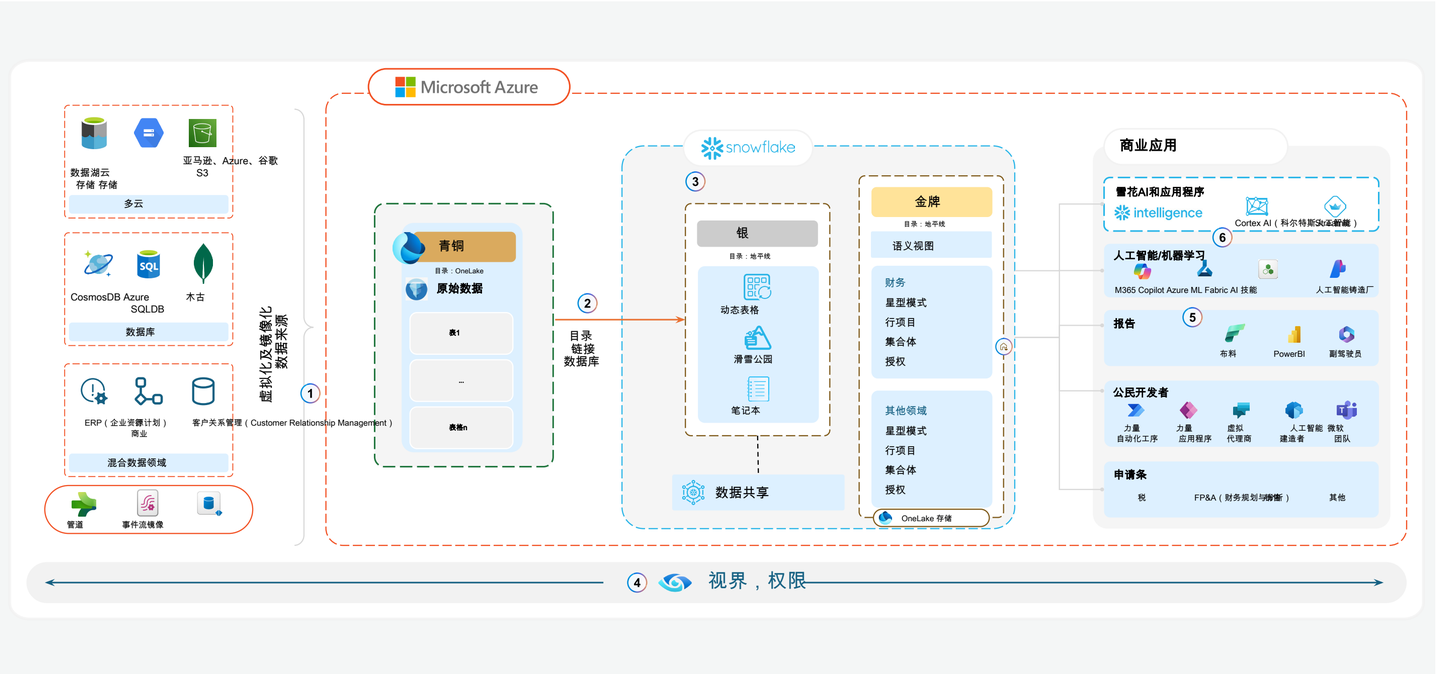


图2：雪晶和微软OneLake的冰山湖屋建筑设计图

1. 使用镜像、快捷方式和集成服务（例如，数据工厂和事件流）将数据导入OneLake。

2. 兼容的、与目录链接的数据库允许Snowflake在OneLake上计算任何Iceberg表。无需ETL或数据复制。

3. 对于已经使用Snowflake本地表的客户，Snowflake将以他们已经熟悉的简单易用体验处理他们的Iceberg表，包括通过AI数据云共享数据来丰富数据或通过Snowflake Intelligence访问它们的能力。

4. 将您的Snowflake Horizon Catalog与Microsoft Purview连接，让您能够扫描、编目和分类您的数据。

5. 直接将Power BI连接到Snowflake数据进行导入、直接查询和直接湖模式。

6. 在所有数据中利用Cortex智能体加速人工智能，直接集成至AI Foundry、Copilot Studio和Azure人工智能生态系统。

人工智能加速了统一整个数据资产的需求，无需通过缓慢、昂贵的文件复制或ETL作业过程。通过开放表格格式和互操作性，客户能够在不增加复杂性或受到供应商锁定的情况下，根据自己的需求灵活设计合适的架构。

—阿伦·乌拉格
总裁，Azure 数据，微软

冰山湖屋配备雪花和AWS

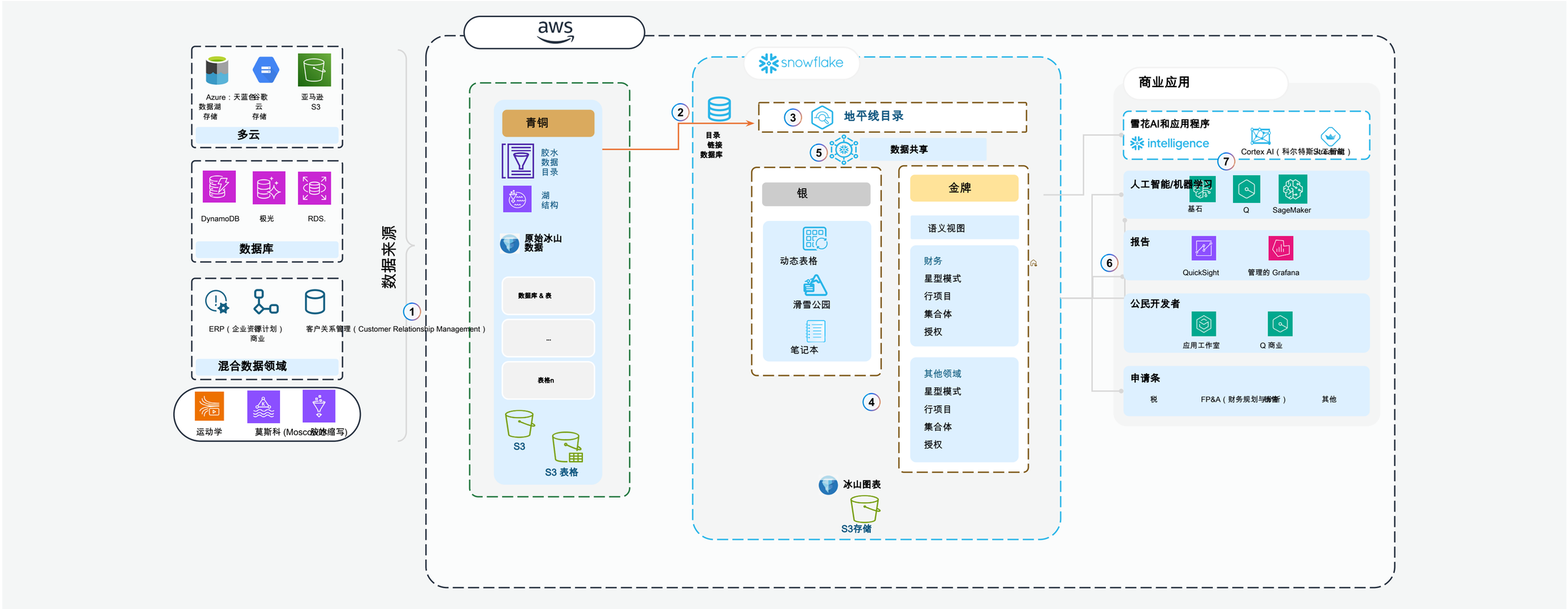


图3：带有Snowflake和AWS的冰山湖屋建筑图

使用Glue、消防水带、DMS、Openflow等集成服务将数据落地到S3和S3表的Bronze层，并由Glue进行分类以及使用Lake Formation进行访问控制。

2. 兼容的、与目录链接的数据库允许Snowflake在Glue目录管理的数据上进行大规模的读写操作，无需数据重复或ETL操作。

3. 雪花水平线目录管理并控制所有雪花中的数据，包括冰山和本地表。它还为胶水目录中的冰山表提供可见性，以方便更容易发现整个数据资产。

4. Snowflake可以处理青铜、银和金层的数据，包括由Glue管理的冰山和S3表、Snowflake管理的冰山表以及本地表，使用高级功能如 动态冰山表格 雪公园和笔记本。

5. 银色和金色等级中的增强数据也可以在AI数据云中复制并安全共享。
6. 将丰富的AWS生态系统服务和其他应用程序连接到Snowflake，以查询和处理精选数据。
7. 利用Snowflake Intelligence、Cortex AI功能和Streamlit加速AI项目，并提供与AWS AI服务（如Bedrock和Q）的代理级别集成，包括Snowflake Cortex代理和MCP服务器。

冰山湖屋，配备雪花和谷歌云

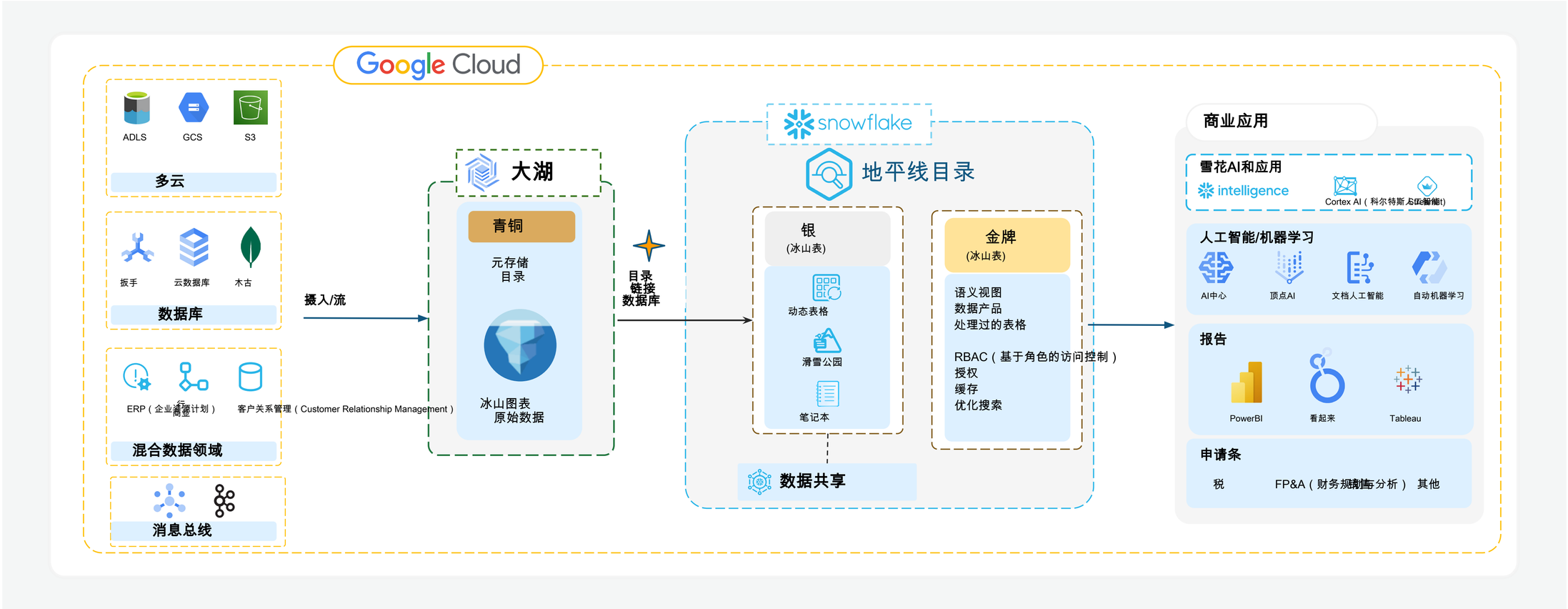


图4：雪晶体和谷歌BigLake的冰山湖屋建筑图。

- *青铜层包括来自不同数据源和OLTP应用程序的历史和未处理数据。在更简单的用例中，Snowflake Horizon可能比BigLake更适合。*

雪花与BigLake的连接（无自定义头部问题）将于2026年4月可用。

- *银层是最有用的地方，因为在数据生命周期中，Spark、Flink、Trino、Dataproc和Snowflake等引擎执行了大量的变换。*

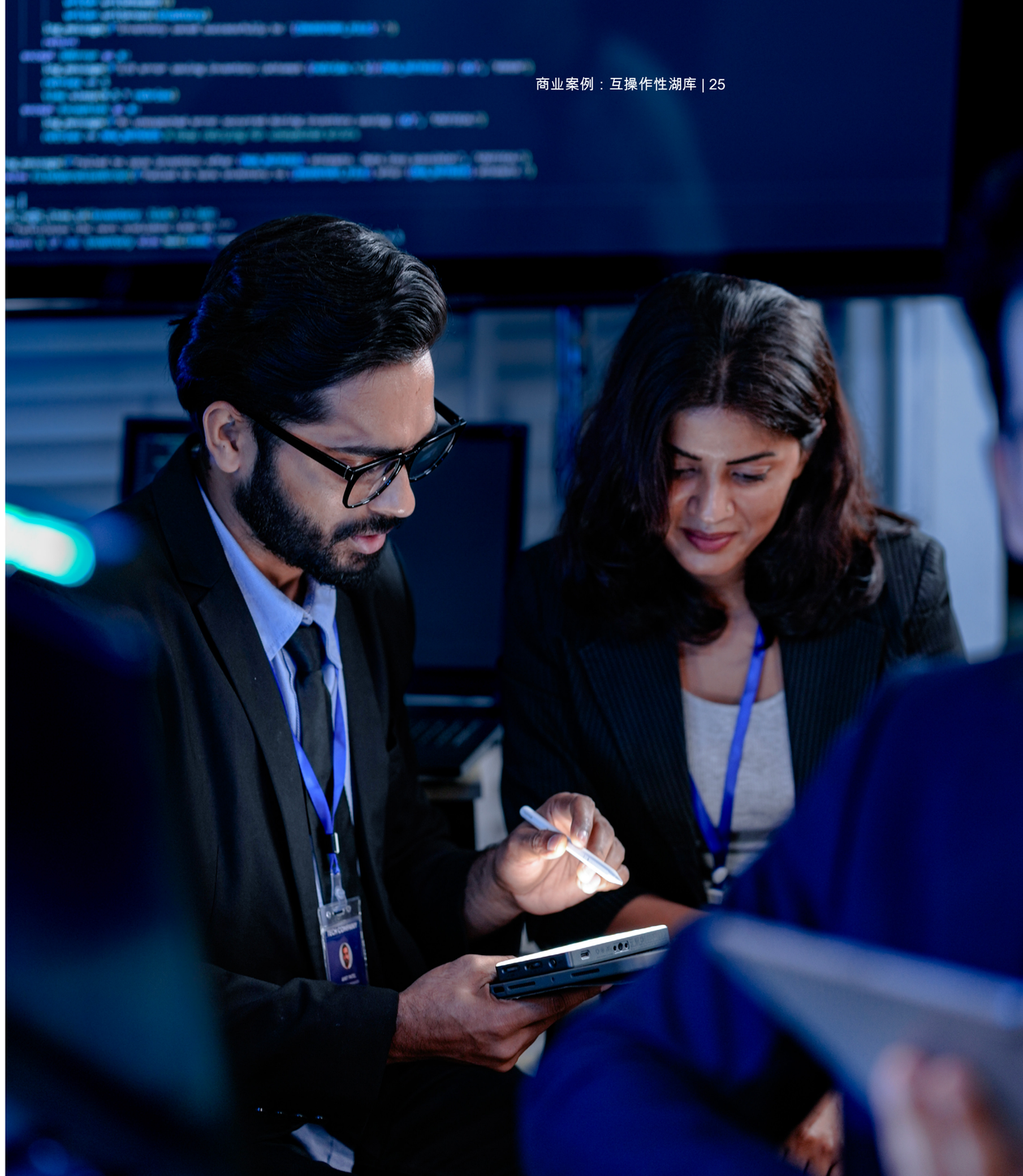
雪花通过直接SQL进行重量级转换，而其他引擎（Spark、Dataproc、Flink等）可以通过IRC读取/写入Iceberg表。

商业案例：可互操作的数据湖

随着组织超越AI实验进入实际部署阶段，许多人发现他们最大的限制并非模型，而是他们依赖的基础数据架构。多年在旧基础上叠加新工具，已形成既强大却又缺乏AI所需的治理和连接基础的大厦。它们僵化且孤立，难以治理且维护成本高昂，但湖屋可以为AI提供更好的基础。

在实践中这看起来是什么样子？在这里，我们分享了一些组织采用Snowflake互操作湖屋策略的真实世界故事。以下每位客户都有Snowflake部署，支持Iceberg用例，适用于所有三家云提供商——微软Azure、AWS和谷歌云，这进一步强调了真正互操作性的重要性。

看看他们的故事，你会看到一个模式浮现：更快地获取高质量数据，更简单的治理，以及消除阻碍进步的日常摩擦的实质性减少。这不仅仅是抽象的承诺，而是具体的成果——效率提升、更明智的决策和从想法到影响的AI倡议。 关于建筑框架，而非尽管如此。 *因为*



Goldman Sachs

像许多大型全球机构一样，高盛面临着扩展挑战：在复杂的环境中管理超过900个不同的数据来源。遗留的数据孤岛和重复的管道导致了数据解释的不一致，从原始供应商数据到见解的转变需要缓慢的、超过15天的周期。为了现代化，该公司需要一个结合直接访问数据提供的灵活性，同时在不影响性能的前提下保持严格的治理，因为该公司处理的是高度监管的金融数据。

解决方案：传说湖屋

高盛 建造了传奇湖屋 一个基于Snowflake和Apache Iceberg统一架构的开源数据平台。

- 互操作性：通过将Iceberg作为其架构的连接组织，高盛从供应商中立的元数据管理和跨之前分散的数据领域中的“整洁连接性”中受益。
- 自动工程：GS 采用 Snowflake 动态冰山表格 为了自动化复杂的转换，用简化的声明式管道替代手动编辑。根据公司所述，这一变化显著提高了某些工作负载的查询性能，在Dynamic Iceberg Tables上的处理速度提高了至多77%，且维护这些管道所需投入的工程时间更少。
- 自助治理：一个集中的“工作台”为数据工程师提供了一站式服务，以便管理审计、使用报告和数据质量保障。

高盛将15天的数据处理周期缩短至一天。

通过结合Iceberg强大的互操作性和供应商中立性以及Snowflake的处理能力和易用性，高盛将那15天的数据处理周期缩短到了一天，成功地为企业提供了更快的洞察力和更强的控制力。

观看完整视频

作为一个快速增长的“先买后付”服务提供商，Affirm面临着处理庞大金融数据的挑战。其传统的复制系统依赖于缓慢且成本高昂的每日快照导出，处理时间长达两到三小时。这种延迟，加上缺乏可靠的“一次处理，永不重复”处理，威胁到了公司数据的完整性。

解决方案：一个开放、互联且可扩展的金融数据基础

肯定 现代化了其建筑 通过采用Apache Iceberg和Snowflake为Apache Polaris提供的托管服务。这种组合使得Affirm能够在使用Snowflake的托管查询引擎的同时保持数据主权。

- 管道效率：通过用高性能变更数据捕获(CDC)管道替换缓慢的每日批量导出，Affirm将其复制过程的月度服务费用降低了6倍。
- 数据新鲜度和可靠性：通过使用Iceberg分支实现的“写-审计-发布”模式有助于确保只消耗经过验证的数据。结合Iceberg的精确一次处理MERGE功能，这提高了团队满足数据可用性服务水平的能力，提高了50-66%——将数据处理延迟从三小时缩短至不到60分钟。
- 审计就绪的时间旅行：Affirm利用零拷贝Iceberg标签，为财务审计和报告提供高效的时间旅行功能。

> **“凭借高性能的CDC管道，Affirm实现了每月服务成本的6倍降低。”**

通过与Snowflake的更紧密集成，Affirm获得了实现核心业务——以完美精确度服务数百万贷款所需的运营简便性。

观看完整视频

indeed

将数百万求职者和雇主连接起来是一项巨大的任务。然而，对于 Indeed 来说，这是公司的首要任务。尽管如此，其基于 PostgreSQL 的基础设施却无法处理其不断增长的 52 个皮轉数据湖的关键任务报告和分析。这导致严重的数据工程瓶颈，使得团队被积压的任务压得喘不过气来，阻碍了分析师及时获取决策所需的数据。

解决方案：一次编写，处处可读的架构

的确 转型为一个现代化、可互操作的湖屋
通过利用Snowflake对Apache Iceberg的原生支持进行架构设计。此次集成解决了遗留瓶颈，同时实现了即时性能和财务收益：

- 开放且互操作：为确保长期灵活性，Indeed选择Apache Iceberg而非Delta Lake和Apache Hudi，以建立与其首选引擎的“一次写入，处处读取”模式。这种方法提供了管理海量数据所需的表现力、安全性和可扩展的计算能力。

- 平台现代化与成本效益：确实已将其整个数据湖转换为Apache Iceberg，允许分析师通过Snowflake直接对湖进行读写操作。

在内部测试期间，公司报告称，使用Snowflake查询这些冰山表比某些替代工具更节省成本，效果为43-74%，具体取决于工作负载和查询类型。

- 民主化的自助服务：新的架构打破了数据孤岛，将Indeed的数据平台（由Snowflake提供动力）变成了一个单一的真实信息来源。通过赋予分析师自助服务功能，Indeed消除了工程积压，并在不牺牲严格的安全和治理标准的情况下，加速了企业范围内的决策过程。

- 增强的可扩展性：摆脱传统系统使得Indeed能够实现其在全球平台上进行大规模关键任务报告所需的高性能和安全保障。

确实报道说，使用Snowflake查询Iceberg表比以前的方法节省了43-74%的费用。

阅读完整客户故事

结论
建筑的力量

如果从这些页面中有所收获，那就让它成为这个：基于Apache Iceberg的互操作性湖屋不仅使您的数据栈现代化，它还重塑了您的企业可以利用数据做的事情。它用互联、开放和互操作的基石取代了庞杂的系统，最终实现了开放表格格式的承诺。这里所强调的组织并非例外——它们是当数据架构不再成为限制而开始成为催化剂时可能性的先驱。

现在，想象一下一个互操作性的湖屋如何提升您的优先事项。探索Apache Iceberg如何简化管道。考虑更强大的通用元数据驱动治理如何解锁停滞的人工智能项目。再想想您的团队在这样一个旨在与他们一起扩展的架构下，可以完成什么，这个架构通过提供在他们的首选引擎和工具中工作的灵活性，而无需数据迁移。

当你准备好的时候，深入探索Snowflake的 互操作性湖屋资源 与我们的专家交谈或开始试点您的第一个用例。通往人工智能成功的道路比以往任何时候都更加清晰——机会也从未如此深远。

雪花是AI时代的平台，使企业能够更快地进行创新，并从数据中获得更多价值。全球超过13,300家客户，包括数百家世界顶级公司，使用Snowflake的AI数据云来构建、使用和共享数据、应用程序和AI。有了雪花，数据与AI对每个人都是变革性的。

了解更多信息，请访问snowflake.com。

（纽约证券交易所：SNOW）