当大语言模型猜测数据模型时，谨防陷阱：以粒度感知防护机制避免文本转 SQL 中的扇出及其基准测试

Ahmed M. Mahran
Incorta 埃及
ahmahran@gmail.com
ah
med.mahran@incorta.com

arXiv:2611.29009v1 [cs.DB] 21 Nov 2026

摘要

基于大语言模型（LLM）的文本转 SQL 系统经常在一对多和多对多连接上进行聚合，这是一种经典陷阱：连接后的行重复了某个实体时，求和或计数会将其计算多次，但查询仍能运行并返回一个看似合理的数字。我们用聚合所读取关系上的一个条件来刻画这类扇出错误；该条件取决于模式中未体现的两个事实：每个度量的粒度，以及连接在每个键上会重复出多少行。通过在数据上检验该条件的审计发现，广泛使用的基准测试 BIRD 的开发集问题中有 4.2% 的标准答案失真；而 BIRD 后续的独立修订中，审计标记为确定扇出的答案有 44% 被修改，相比之下，未标记答案的修改比例为 16%。我们发布了 BIRD-FanOut（包含 486 个经审核标准答案的 BIRD 问题）和 GrainTraps（一组设置此类陷阱及相关陷阱的合成问题）。在八种大语言模型智能体的测试中，基于数据计算出的函数依赖（FD）在 GrainTraps 上并不比逐列统计信息更有帮助；但同时说明每个度量粒度和每个键上的重复情况的粒度感知函数依赖，能将 BIRD 上的扇出错误从占运行次数的 18.4% 降至约 13%，这一效果在留出问题上同样成立，而且不需要大语言模型评审器。准确率提升较小，因为剩余错误大多并非扇出错误，而且 BIRD 的提示往往会引导模型过度计数。应向文本到 SQL 模型提供数据的粒度，并依据数据审计基准测试的标准答案。

代码、数据和基准测试可在 https://github.com/ahmed-mahran/misqal 获取

1 引言

一位分析师向文本转 SQL 助手询问：“包含玩具的订单的总运费是多少？”自然的查询会将订单与其玩具明细连接，筛选出玩具明细，并对运费求和（图 1）。查询能够运行并返回一个看似合理的数字，但结果是错误的。运费在每个订单中只存储一次，而连接会使其在每个玩具明细中重复出现，因此包含两条玩具明细的订单会被计费两次。错误源于查询从未检查过的一个数据假设，即每个订单在待求和的行中只出现一次。

这是我们称为 扇出错误 的一大类错误中的一个实例。当查询对实体重复出现的行进行聚合时，重复敏感聚合（计数、求和、平均值或 ORDER BY ... ... LIMIT）会将该实体视为出现了多次。扇出错误是主要的

一种是粒度错误：在不同于问题所需粒度的层级上计算聚合；另一种是在错误的粒度上分组或排名，且没有任何重复（§2.1）。重复有几种来源：非唯一键连接（典型的扇出和断层陷阱）、在每个订单行中重复订单级值的反规范化表，以及多对多映射。数据仓库通过以下方式防范

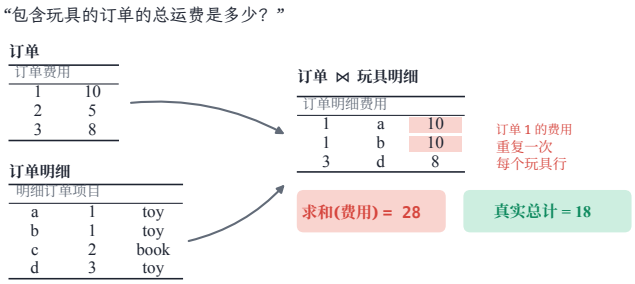


图 1：一个扇形陷阱。问题需要通过订单明细来筛选订单，但费用是按每笔订单存储一次的；连接会使其在每条玩具明细行中重复一次，因此求和时会将订单 1 的费用计算两次（28，而不是 10 + 8 = 18）。

这些问题：声明每个事实表的粒度，也就是每一行代表什么（一个订单、一条订单行）[34]；语义层会声明连接基数 [12, 18, 23]，但无法描述单个宽表内部的依赖关系。面对原始数据库，文本转 SQL 模型既没有这些信息，也只能看到表名和列名，因此必须猜测或检查其背后的数据模型。

无论对模型还是对评估而言，扇出都难以察觉。 扇出错误不会引发异常，通常还会返回一个看似合理的数字。评估很少能发现这类错误。BIRD [40]，是一个广泛使用的文本转 SQL 基准测试，包含 95 个真实数据库中的 12,751 个问题；它通过基于集合的执行准确率（EX）对答案评分，而这种指标会忽略结果中的重复行，因此掩盖了开发集上 5.8–6.8 个百分点的多重性错误 [66]。改变聚合值的过度计数确实会导致 EX 未通过，但前提是标准答案正确；而 BIRD 的标准答案本身也存在扇出问题：4.2 % 的开发集问题受到失真影响，清理后的发布版本仍存在确定的扇出标记（§4）。提供基数信息的系统报告称，其影响较小（移除此信息会使 MCI-SQL 的 EX 降低 0.5 个百分点 [60]，使 SQLfuse 降低 2 个百分点 [64]），但这些影响是通过 EX 而非扇出错误来衡量的。

本文。 文本转 SQL 模型需要两个模式未展示的事实：每个度量的粒度，以及连接在每个键上会重复多少行；这两者都可以从数据中计算得出。我们提出三个问题。BIRD 的标准答案中存在多少扇出，基于数据进行测试的审计能否发现这些问题？当前的大语言模型何时会产生扇出错误？从数据中计算出的证据是否能帮助它们避免这些错误，尤其是那些说明每个度量粒度以及连接在每个键上会重复多少行的证据？每个模型都作为智能体运行，通过 SQL 工具查询数据库，然后提交一个最终查询。我们比较七个

实验组，各实验组之间的差异仅在于模型接收到的证据不同。图2展示了本研究。

贡献与发现。 本文提出了四项贡献，并分别给出主要发现：

1. **聚合的粒度条件** (§2.2)。当某个度量粒度上的每个实体在其读取的关系中恰好出现一次时，对重复敏感的聚合才是正确的；其误差等于各实体的度量值乘以该实体超出一次的多重性。这是联机分析处理可汇总性[39] 的一种查询级形式，适用于任意 SQL 聚合所读取的关系，并且可以在数据上进行检查。它明确指出模型必须掌握的两个事实：度量粒度，以及连接所导致的多重性；并将扇出、断层和多对多陷阱视为违反范式的情况。

2. **扇出审计与 BIRD-FanOut** (§4)。该审计针对数据，并在每个查询自身的筛选条件下，检查聚合读取的行是否存在重复。扇出导致 BIRD 开发集的标准答案中有 4.2% 失真；训练集金标准中有 6.2% 带有明确标记；BIRD 的独立修订将审计标记为确定存在扇出的标准答案中有 44% 作了修改，而未标记的标准答案中这一比例为 16%。BIRD-FanOut 包含 486 个 BIRD 问题，其标准答案经过标注、修正，并由大语言模型评审器审查。在模型答案上运行时，该审计能标记出大多数计数过高的答案，也很少误标正确答案，但无法判断问题是否本就要统计这些重复项(§6.7)。

3. **GrainTraps** （§5.1）是一个独立于 BIRD-FanOut 的合成测试套件：涵盖三个生成的数据库中的 59 个问题，以明确和模糊两种表述形式隔离出六种陷阱。没有证据时，每个模型都会在其中出现粒度错误，在 2% 至 42% 的运行中如此。回答前会检查数据的模型出现的错误最少，但要求一个模型进行检查，只会让它检查得更多，并不会减少错误（§6.1）。

4. **一项关于数据衍生证据的对照研究** （§5.2、§6）：涵盖八个大语言模型、七个实验组和 38,232 次智能体运行。在 GrainTraps 上，“可能有多个”之类的函数依赖标签并不比逐列统计信息更有效。粒度感知函数依赖会说明每个度量的粒度、按每个键连接会重复多少行，以及空值会在何处打破依赖关系；它将 BIRD 训练集上的扇出错误比例从无证据时的 18.4%、使用统计信息时的 17.0%，降至 13.1%–13.6%，且在 183 个留出问题上也能保持这一降幅；仅重复计数就能带来约一半的降幅。准确率仅提升约 3 个百分点，因为 BIRD 的提示往往会指定失真公式，而轻微的过度计数在数据中几乎不显眼（§6.4）。

2 背景

2.1 粒度错误及其陷阱

图3概述了查询可能重复计数某个实体或在错误粒度上计数的六种方式。扇出和断层陷阱是实体关系建模中的连接陷阱，即实体之间存在有歧义或断裂的路径 [17, 30]；商业智能工具用这两个名称指沿此类路径连接所产生的重复计数 [12],，我们则将这些名称扩展到另外四种情况。度量是查询会进行聚合的数值列（例如费用、数量）；其粒度是它对每个实体只有一个值的那个实体（例如运费对应一个订单，数量对应一行）。一个问题如果存在陷阱，通常也会有一个陷阱查询，即自然会落入该陷阱的 SQL 查询。

- **扇出**。 无论是通过连接引入子行，还是反规范化表中已包含这些子行，父级度量都会对每个子行重复一次：订单的运费会按每行计数（图1）。

- **断层**。同一父实体的两个独立子实体通过它连接后会彼此相乘：每条产品订单行会按库存批次重复一次，而每个批次也会按行重复一次。

- **多对多映射**。 桥接表会使度量按每个关联实体重复一次：产品的价格会按每个标签计数一次。
- **先排名后关联**。如果排名的分组粒度比问题要求的更细，前N名就会排错对象：问题要求找出前五名产品及其按客户划分的销售额，查询却返回了规模最大的五个产品–客户对。

- **投影连接**。分组键无法确定的列会拆分或重复行：某个区域的收入会针对该区域的每个目标行重复一次。

- **NULL 分割**。 空值会形成独立分组，并拆分实体：某个产品在部分行中缺少类别时，会出现在两个分组中。

落入陷阱的答案会产生粒度错误：它以不同于问题要求的粒度进行聚合。粒度错误分为两类：

- 扇出错误：行会重复实体，如扇形、断层、多对多和投影连接陷阱，以及反规范化表中的情况（扇形陷阱是连接导致行扇出的最简单方式）；

- 分组错误：粒度不正确，但没有行重复，如先排名后关联和 NULL 划分。

陷阱查询通过构造使其问题产生粒度错误，因此，结果等于陷阱查询结果的答案也会产生该错误；我们将此类错误称为精确错误。若答案还犯了其他错误，则会返回不同的结果，因此§5.1还会统计审计发现的粒度错误，而§6则统计大语言模型评判器发现的粒度错误。

2.2 何时聚合是正确的?

设一个查询范围在关系 $Q = \sigma_\varphi(R_1 \bowtie \cdots \bowtie R_k)$ 上，在多重集语义下对度量 m 进行聚合（如果该范围包含分组，则在每个分组内进行）。其中， m 是某个粒度为 G的 R_i中的列：函数依赖 $G \to m$在 R_i中成立，问题要求针对每个 G值求出 m的一个值。由于输入中的每个函数依赖在选择和等值连接下都成立，因此 $G \to m$在 Q中也成立。令 E为 Q中的 G值集合，并令多重性 $\mu(g)$表示 Q中满足 $G = g$的元组数量。换言之，重复敏感聚合会根据实体的多重性为其加权：

命题 *2.1*（粒度条件）。在粒度 g 上对非 *NULL* $m(g)$求和 $SUM_Q(m) = \sum_{g \in E} \mu(g)\, m(g)$，因此它与按实体求和的结果相差 $\sum_g (\mu(g) - 1)\, m(g)$；计数 $COUNT_Q(m)$ 与非 *NULL* m 实体的数量相差 $\sum_{g:\, m(g) \neq NULL} (\mu(g) - 1)$。因此，当每个此类 g都满足$\mu(g) = 1$ 时，*SUM* 和 *COUNT* 才正确；*COUNT* 仅在此时正确，而非负度量的 *SUM* 只有在 $m(g) > 0$之处， $\mu(g) = 1$ 时才正确。*AVG* 是 m 在 E上的 μ加权均值，当且仅当 μ 在 E上为常数时，对每个 m 都正确。*MIN*、*MAX* 和 *COUNT(DISTINCT m)* 则

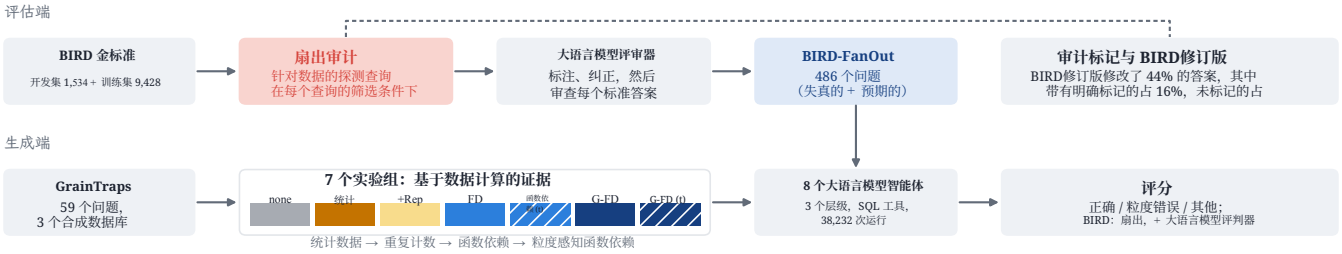


图2：研究概览。上：对数据进行审计，标记出会重复聚合行的 BIRD 金标准查询；大语言模型评审员为这些标记分类并进行修正，BIRD 的独立修订对其进行验证，最终得到 BIRD-FanOut。下：三个层级中的八个大语言模型智能体，在最多七种证据提供方式下回答 BIRD-Fan Out 和 GrainTraps（一个独立的合成陷阱套件；BIRD 开发集上为六种；G-FD：Grain-FD，（t）：通过工具提供）。所有图中均使用这些颜色；带阴影线或空心标记表示通过工具提供。我们统计 GrainTraps 上的粒度错误（答案复现陷阱查询的结果，或审计将其标记出来），以及 BIRD 上的扇出错误（精确匹配，或由大语言模型判定；§6）。

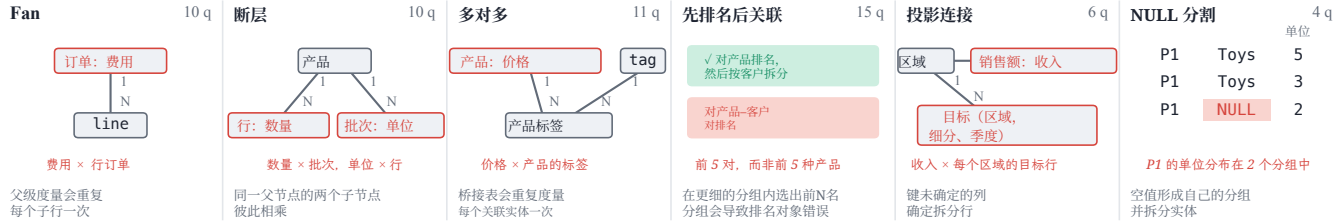


图3：导致粒度错误的六种陷阱；除先排名后关联和 NULL 分割外，其余均为扇出。红色标出重复的度量或被拆分的实体，斜体行说明重复或拆分的幅度；对于先排名后关联，绿色框表示正确的操作顺序，红色框表示陷阱。每个面板右上角的数字表示该类型的 GrainTraps问题数量（§5.1），包括明确和含糊的表述；未显示对照项。

不依赖于 μ，而且对失真的聚合执行 *ORDER BY . . . LIMIT* 时，会按失真的值进行排名。

§2.1 中的多重性陷阱，是指 μ 超过 1 的各种情况。在扇形陷阱中，$\mu(g)$ 是父实体 g 的子行数量。在断层陷阱中，父实体 p 的两个独立子实体 S 和 T 会导致度量 S 乘以多重性 $|T_p|$，而父实体的度量 $|S_p|$ 则乘以 $|T_p|$（没有子行的父实体会被排除）。桥接表对每个关联实体贡献一个元组，投影连接对附加列的每个值贡献一个元组，而反规范化表无需连接，其 $\mu > 1$ 会直接存在。假设没有基表包含重复行，$\mu(g) > 1$ 表示 G 不是 Q 的超键，尽管 $G \to m$ 在 Q 中成立，因此 Q 违反 BCNF：扇出是聚合所读取关系的规范化失败；扇形陷阱中存在传递依赖（行 → 的订单 → 费用），而断层陷阱中则是独立多值事实的乘积。先排名后关联和 NULL 划分属于分组粒度错误，而非多重性错误，因此不在该命题的讨论范围内。该条件针对聚合所读取的关系，而非答案：用户会询问细分结果，而 GrainTraps 的 32 个明确标准答案中有 9 个（包括对照项）不符合 BCNF。对于答案，我们只要求每个单元格有一个值，这也是提示所要求的（§5.2）。

该命题是联机分析处理中可汇总性在查询层面的对应概念 [39, 43], , 其条件（互斥性、完备性、类型兼容性）是针对已声明的维度层级结构提出的。

这里，条件针对任意 SQL 聚合所读取的关系提出，即经过连接和筛选条件处理后的关系；其两个输入都可以在数据上进行度量：每个度量的粒度 G，以及查询中的连接和筛选条件所导致的多重性 μ。模式中看不到这两者。审计（§4）会在数据上检查：在每个表的行粒度下，应用查询自身的筛选条件后，$\mu > 1$。粒度感知的函数依赖证据（§5.2）会为模型提供这两个输入的逐表形式：每个度量的粒度，以及对于每个标识符，每个值会出现多少行；这个数量就是基于该标识符进行连接时匹配行被放大的倍数。模型仍需在各个连接之间综合这些信息（§2.3）。诸如“每个订单_id 可以对应多个行_id”的函数依赖标签，只表明 μ 可能大于一，却没有说明具体大多少，也没有说明针对哪个度量。

2.3 实例级函数依赖关系

我们的依赖关系证据（§5.2）是根据数据计算得出的，因此我们使用实例级函数依赖（FDs）。对于表A、B两列，若表r满足r，则$A \to B$当且仅当$d(A) = d(AB)$，其中d用于计数不同值组合 [7, 31]。我们将 NULL 视为一个值（null= null，即 SQL GROUP BY 的语义）；空值感知函数依赖理论则区分其他概念 [8, 36]。通过比较$d(A)$、$d(B)$和$d(AB)$，可将每对列标记为一对一、一对多、多对一或多对多。若表中存在某函数依赖，则该依赖在任何筛选条件下都成立；若一对列不存在函数依赖（一对多、多对多），则筛选后可能出现函数依赖，因此其警告（“可能有多行”）是保守的。

逐表事实中的表级函数依赖可通过传递性延伸至连接（一个表中的 $A \to id$ 和另一个表中的 $id \to B$ 可推出连接结果中的 $A \to B$）[35, 50],，但非依赖关系不能如此延伸；连接是否会重复行，取决于其键的唯一性。模型只接收逐表事实，必须自行将它们结合起来；阐明这些事实对模型自身连接操作的影响是未来工作（§7）。

3 相关工作

面向大语言模型的数据衍生依赖关系。 MCI-SQL [60] 使用 TANE 挖掘表内成对函数依赖，并通过元数据工具提供这些依赖（移除该工具会使 BIRD Mini-Dev 上的 EX 损失 0.5）。SQL- fuse [64] 使用数据衍生的一对多关系（在 Spider 开发集上获得 2.0 EX [63]）。Shkapenyuk 等人 [56] 提供单列概况；Dex- terSQL [52] 通过度量连接扇出来消除列的歧义；Su 等人 [58] 报告称，带基数标注的键带来的增益很小。我们的函数依赖标签（函数依赖，§5.2）与 MCI-SQL 的成对标签理念相近，但并非复现。不同于这些工作，我们直接度量扇出错误，而不是 EX，并在受控实验组中区分统计数据、重复计数和依赖关系。

智能体探索。 在回答前查询数据库的智能体 [20, 38] 会隐式检查数据；我们度量它们何时会为扇出而这样做。

本体与知识图谱。 基于本体的关系数据访问 [13, 53] 允许用户围绕实体而非表提出问题。Sequeda 等人 [55] 表明，在保险数据库的手工构建本体和知识图谱上向 GPT-4 提问，可将准确率从 16% 提高到 54%；再结合基于本体的查询检查与大语言模型修复，准确率可提升至 72% [3]；英伟达的 Auto Ontology 则可自动构建这样的层 [44]。本体中的函数属性（“一笔订单恰好有一项运费”）是一种声明的依赖关系；将订单建模为独立类别的本体，则能从构造上避免扇形陷阱。我们的证据是本体中这部分内容在实例层面的自动推导对应物，而我们的审计则发挥基于本体的查询检查的作用，通过针对数据而非声明的公理检查查询。

扇出检测与修复。 语义层及其验证器依据声明获取基数：Cortex Analyst 的扇出和断层检查 [12],、Looker 基于声明的主键进行的对称聚合 [24], 、GROUND 声明的粒度 [51], 、dbt 代码检查器 [15]（在 14 个公开 dbt 项目中，765 个连接里有 463 个的粒度对代码检查器而言未知）。DevRev 的大语言模型计划验证器会检查去重放在何处 [59]；ModularSQL 会标记含有大量重复行的结果，并在外层查询中添加 DISTINCT [66],，以至于过度计数的聚合只要得出一个看似合理的数字就能通过；基于规则的双重计数消解器早在大语言模型出现之前就已问世 [29]。粒度理论在类型层级上形式化了粒度 [33],，而可汇总性 [39, 43] 阐明了何时沿声明的维度层次结构进行聚合才是正确的；即时聚合 [14, 62] 则是经典方法。所有这些方法都依赖声明的基数或粒度，或是在不查看数据的情况下修补查询；我们的审计无需任何声明，并会针对数据测试每条查询。

函数依赖发现。 TANE [31], HyFD [49] 以及一项综述 [1] 涵盖了函数依赖发现；Abedjan 和 Naumann 的唯一列发现 [2] 涵盖了我们使用的基于计数的剪枝；Kossmann 等人验证了由查询驱动的依赖关系 [37]。我们并未声称计算这些证据具有新颖性；我们关注的问题是，其中哪种形式有助于模型。

基准测试质量。 Jin 等人 [32] 发现，BIRD Mini-Dev 和 Spider 2.0-Snow 的样本中有一半以上存在标注错误，包括非唯一键连接；sqlsure 报告称 BIRD 金标准[57]的数值被 8× 夸大了；一项从业者分析将某系统在 BIRD 上部分失分归因于对标准答案的过度计数 [28]；Wretblad 等人 [61] 在一组 BIRD 样本中发现了含噪问题和标准答案。测试套件评估 [65] 会在许多数据库上比较查询与标准答案，因此仍然信任标准答案；而歧义基准测试 [9, 54] 会为一个问题配上多种有效解读，正如我们的预期问题所做的那样。我们衡量 BIRD 标准答案在数据上的扇出，并依据 BIRD 的独立修订对其进行验证。

4 BIRD 中的扇出

我们在数据上审计每个 BIRD 标准答案查询，将审计结果与 BIRD 后续自行修订的版本进行核对，并根据这些标记构建 BIRD-FanOut（图 4）。

审计。 审计会测试聚合读取的行是否发生重复（§2.2）。它将每个查询解析为作用域（使用 SQLGlot [42],SQLite 方言）：公用表表达式、子查询以及集合操作的各个分支。对于包含重复敏感聚合或 ORDER BY . . . LIMIT 的每个作用域，在该作用域自身筛选条件下运行探测查询，比较连接后的行数与每个表的不同的行数（SQLite rowid）。如果某个连接可能导致行重复，但在当前数据中并未重复，则不会标记。例如，BIRD 训练集问题9279 的标准答案（图 5）统计连接到客户表的租赁记录中的 customer_id，因此一名不活跃客户会按每条租赁记录计数一次（15 名客户共有 404 条租赁记录）；审计将该 COUNT 标记为确定的扇出。

被标记的查询会获得以下三种标记之一：

- 确定：行在求和、平均值、COUNT(col) 或 ORDER BY ... LIMIT 下重复，或在所有连接的表都存在重复时，出现在 COUNT(*) 下。无论问题的含义是什么，答案都是错误的。
- 有歧义：COUNT(*)，其中只有部分表存在重复。它会精确统计其他表中的行，因此只有当问题统计的是存在重复的表时，答案才会错误。
- 重复值：COUNT(col)，其中该列在其所属表中有重复值。

审计 BIRD 开发集约需 16 秒。在单个反规范化表中，它只检查 COUNT(col)；对连接子查询或 CTE 的情况只进行静态检查；而对于任何重复情况都会标记平均值，即使行数均匀重复时，这种做法也较为保守（§2.2）。

患病率。 在 2024年6月的开发集上，5.3% 的标准查询带有明确标记。两位评审员（Claude Opus 5.5 [5], GPT-5.5 [47]）将每个明确或有歧义的标记判定为失真 （问题指的是不同实体）、预期 或不明确，编写必须能够运行并通过审计的更正后 SQL，并判断能否根据数据模型确定预期解读。两位评审员对 95% 的标记意见一致；一个 Claude 智能体阅读 20 个随机标记后，与每位评审员分别有 18 个意见一致。根据这两项标签，一个带标记的标准答案是

- 失真，若两位评审员都如此标注，则修正其标准答案；
- 预期，若两位评审员都将其标注为预期，则计数重复行是正确的；
- 若被排除 的原因是评审员意见不一致（有争议）、修正后在这份数据上得到相同答案（无害），或该解读无法确定。



图 4：从 BIRD 标准查询到 BIRD-FanOut：10,962 条标准查询、668 个确定标记、486 个问题。开发集标签涵盖 2024年6月 发布版本中的确定标记和歧义标记；开发集候选项来自对 BIRD 清理后的发布版本进行标注。被排除的问题包括有争议、无害或无法解决的问题；在标准答案审核中，还排除了存在争议或没有扇出信号的问题。

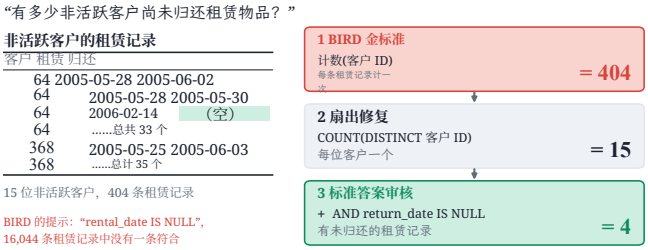
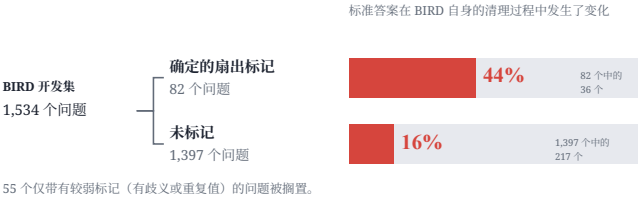


图 5：BIRD训练集问题9279。（1）标准答案对租赁记录进行计数；审计将其标记为确定的扇出。（2）扇出修正对客户进行计数。（3）标准答案审核添加了缺失的“未归还”条件。（空）表示 NULL。

在开发集上，4.2% 的标准答案存在失真；当答案为一个数字时，失真标准答案是修正后答案的中位数 6.3 倍。在训练集上，6.2% 的标准查询带有明确标记，其中失真查询的结果是修正后答案的中位数 1.6 倍。

基于 BIRD 修订版进行验证。BIRD清理后的开发集（2025年11月6日）[10] 保留了问题ID，并在人工审核后重写了标准查询和问题，且独立于我们的审计。对于带有确定标记的问题，其中44%的标准答案发生了变化；而无标记的问题中，这一比例为16%（图6）。在有标记的9个数据库中，带标记的问题也都更常发生变化。该版本还显示了审计发现了多少真正的扇出修复。在两位评审员都认定为保留问题含义的扇出修复的36项答案变更中，审计标记了32项。我们只在添加 DISTINCT 的变更中查找遗漏的修复，因此实际召回率可能更低。清理后的开发集仍有约



55 个仅带有较弱标记（有歧义或重复值）的问题被搁置。

图6：针对 BIRD 独立修订版的审计。在带有明确扇出标记的开发集标准答案中，BIRD清理版（2025年）在未使用我们的标记的情况下修改了44%；未标记的答案中则有16%。

30项被我们的标签判定为失真的标准答案；2026年 Mini-Dev [11] 没有明确标记。

BIRD-FanOut。 BIRD-FanOut保留了失真问题和预期问题。开发问题来自清理后的发布版本；如果该版本修正了扇出问题，其 SQL 就作为标准答案，而旧标准答案则作为陷阱查询。在训练集上，两名评审员（Gemini 3.6 Flash [27], Claude Haiku 5.5 [4]）根据与开发集标签的一致程度选出，并为每个明确标记贴标签；两名强评审员（ClaudeOpus 5.5、GPT-5.6Terra [48]）则重新标注前两名评审员无法确定的项目。只有当两条修正后的查询返回相同答案时，才会保留失真问题。

标准答案审核。 修正会消除扇出问题，同时保留 BIRD 查询的其余部分，因此标准答案可能仍会因其他原因而出错：在40个模型答案中，两名评审员都认为它们是合理解读、但与修正后的标准答案不符；Claude 智能体查询数据后发现，其中21个标准答案有误（否定筛选条件、照搬提示中的拼写错误、使用错误键进行连接）。在9279中，标准答案也完全没有检查租赁记录是否尚未归还（BIRD 的提示将此条件放在 rental_date 上，该字段永远不会为空）；审核后的修正将答案从15降至4。因此，BIRD-FanOut的每个标准答案都会经过审核。Gemini 和 Haiku 会结合提示、模式、示例行以及标准答案的结果阅读每个问题；任何一方标记的问题都会交由强评审员处理。若两位强评审员意见不一致，他们会在不知道对方身份的情况下查看对方的审核意见一次。只有当两位强评审员都认定标准答案有误，且他们修正后的查询返回相同结果时，修正才会被接受；修正必须能够运行、改变标准答案的结果，并且在失真问题上通过审计；若修正返回空结果、0 或 NULL，也会对照数据进行检查。每位评审员还会编写一个陷阱查询，只有双方同意时才予以保留，以便无需评审员即可评估精确扇出错误。因此，BIRD-FanOut共有486个问题，其中295个失真问题，191个预期问题。附录D介绍审核结果、对数据的检查及其对结果的影响。

5 GRAINTRAPS 与证据

5.1 GrainTraps

BIRD 展示了扇出出现的频率；GrainTraps 展示了模型何时会产生粒度错误。它使用三个可复现生成的合成 SQLite 数据库：预订数据库（一个宽型销售表；产品在多个部门中预订，并由多个客户预订）、供应数据库（订单和库存批次，每种产品都对应多个订单和多个库存批次），以及商店数据库（一个宽型订单行表，其中订单级字段在每行中重复；某个属性存在 NULL 间隙、多对多标签，以及按区域、客户细分和季度划分的销售目标）。其中包含 59 个问题：29 个

明确的陷阱问题、27 个模糊孪生问题和 3 个对照项（图3 按陷阱类型列出的问题数量）。

明确问题会说明预期的粒度；模糊孪生问题则在措辞中省略粒度，但保留返回列清单，并说明为何其解读符合数据模型。每个问题都附有书面的陷阱解释和解答、标准 SQL、陷阱查询（即重现该陷阱的朴素 SQL），以及（针对其中一个问题）另一种可接受的解读。NULL 拆分类标准答案将空缺视为产品单一类别的缺失值；将 NULL 报告为单独类别则是陷阱解读。

评分。答案若

- 正确，则其行作为多重集与标准答案或替代答案匹配（浮点数保留 2 位小数，不考虑顺序）；
- 若答案精确，或审计（§4）发现其中存在实际扇出，则属于粒度错误；
- 否则若答案错误，则属于其他；• 若未生成可运行的 SQL，则属于失败。

模型若在提交前结束，则按其最后运行的查询评分；在 §6 的运行中，任何模型和实验组出现这种情况的比例最多为 4.0%。

角色与构建。 我们将该套件用作诊断工具，而非排行榜，因为完整模式运行（§5.2）中的四个较大模型无需帮助即可解决其中 92% 的问题。Claude 智能体在作者指导下生成了数据库、问题、标准答案和陷阱查询。测试会检查数据，确保每个标准答案都返回行，且每个陷阱查询都返回不同的结果；模糊问题只有在强模型能够按原文解决时（在没有证据的运行中）才会保留，因此对于强模型而言，证据几乎没有发挥作用的空间。不存在留出划分；三个对照项都很简单，而有意设计的扇出（故意对连接后的行进行计数）则由 BIRD-FanOut 的预期问题覆盖。

5.2 证据

智能体与实验组。 每个模型都作为智能体运行：它可以根据需要通过工具在数据库上运行 SQL（最多 50 步），最后提交一条 SQL 查询和简短答案（附录 B）；我们将一次运行的记录称为其 运行轨迹。所有实验组共用一个提示，其中写道：不要根据表和列的名称或看似常见的情况，假设它们之间的关系：请在数据中进行检查，并要求仅返回请求的列，每个单元格包含一个值。提示中不包含任何操作流程、陷阱名称或布局提示。七个实验组之间的唯一区别在于模型获得的证据，图 7 原样展示了每个实验组针对一个 GrainTraps 问题向模型提供的内容（附录 A 展示了第二个问题）。统计是内容对照项：每个非度量列的行数、不同值和 NULL 数量，但不关联各列。Stats+Rep 是一个探索性对照项，用于隔离重复计数。证据可以在提示中提供，也可以通过工具提供；该工具是基数_矩阵，模型调用时需指定表和列，并且必须针对其最终查询读取的每个表调用一次。 本节其余部分将介绍所有实验组共用的设置，以及证据的计算方式。

已知表。 对于每个问题，在每个实验组中，智能体只能看到其参考查询（修正后的标准查询和陷阱查询，或原始标准查询）所读取的表及其所有列，而无法看到整个数据库。保留两个查询涉及的表，可使两种解读都保持开放。在 BIRD 训练集上，这会将模式缩减至原大小的 27%。

完整模式运行。 另一组运行则向智能体展示整个数据库，并将度量列纳入证据：在 GrainTraps、BIRD-FanOut 开发集以及 150 个普通 Mini-Dev 问题上测试 Claude Opus 5.5、Claude Sonnet 5.5、GPT-5.5、Gemini 3.1 Pro [25], Gemini 3.5 Flash-Lite、GLM、GPT-OSS-120B [46]（未能完成主要运行）和 Nemotron（2026年发布版，包含连接且没有审计标记）。§5.1、§6.1、§6.6 和 §6.7 使用这些运行；附录D列出了各实验组及主要结果。

列角色。 在提出任何问题之前，已知表中的每一列（352 张表中的 2,879 列）都会被标注为标识符、维度 或度量，如同数据目录中的标注。角色由启发式方法（键、类型、数值占比、BIRD 的列描述）和每张表的两位大语言模型评审员（Gemini 3.6 Flash、Claude Haiku 5.5；两者有 97% 的一致率）共同判定，剩余列则由 Claude Sonnet 5.5 判定；其中 704 列为度量。仅靠启发式方法会将 613 个整数列列为未定，并将 116 个以文本形式存储的数值列误判。列角色决定证据涵盖哪些内容。度量不会纳入统计数据和依赖关系概况，因为对它们进行建模会引入噪声，例如偶然出现的键（“unit_price 是一个键”，因为所有价格都不同）；Grain-FD 则会说明每个度量的粒度。标识符会统计重复次数。模式不会显示列角色；模型只有在证据明确指出某列是度量时，才能得知这一点。这些角色信息随基准测试一同提供。

FD 证据。 对于一个表及其非度量列，聚合 SQL 会为单列和列对计算$d(\cdot)$，并剪除其标签可由其他标签推导出的列对（一对一替换、传递性、基数界限、采样反例） [2]。该概要会报告推断键、1:N 层次结构以及每个列对的标签（图 7）。

Grain-FD 证据。粒度感知证据在 FD 证据的基础上增加四项内容，为 §2.2 提供其所需的两个输入。它针对每个表计算一次并进行缓存，以与 FD 证据完全相同的方式交付，因此 FD 与 Grain-FD 以及 FD（工具）与 Grain-FD（工具）的区别仅在于内容不同。这些示例来自图 7 所示的表。

- **度量粒度**量排除在依赖关系概要之外（如上所述），也会掩盖这样一个事实：运费会在其订单的每一行重复出现。Grain-FD 会说明每个度量的粒度：度量具有单一值的最粗非度量列（“运费_：每个订单_id 一个值（平均每个订单有 3.5 行；对行求和会将其计算 3.5 次）”）；如果不存在能够确定该度量的单列，则说明该度量随行变化。
- **重复计数**“可能有多个”表示连接可能会重复行，但没有说明重复的频率。Grain-FD 为每个标识符补充每个值对应的行数及其后果：“订单_id：1–6 行（平均 3.5 行）；基于该字段的连接会将每个匹配行重复相应的次数”。
- **感知 *NULL* 的函数依赖**函数依赖会将 NULL 计作一个值（GROUP BY 语义），因此只要有些行的类别为空值，“每个产品都有一个类别”这一说法就不成立，标签也会显示“可能有多个类别”。Grain-FD 只在已填充的行上重新检查每个不成立的函数依赖，并报告差异：“product_name →category 在已填充的行上成立；523 行的类别为空值”。



图 7：针对一个 GrainTraps 问题（“每个客户细分的运费总额是多少？”，表 sales_lines），各实验组向模型提供的内容，摘自运行轨迹并经删节（. . .）；已去除 Markdown 标记，并将多对多箭头绘为 ↔。蓝色：某实验组相较于其参考组新增的内容；灰色斜体：模型自身的工具调用。只有 Grain-FD 实验组指出，运费会在该订单的 3.5 行中重复出现。

- 范围限定。 Grain-FD（工具）始终会将表的标识符添加到请求的列中（最多 25 个），因此范围限定调用不会遗漏承载度量粒度的键。

设计集与留出问题。 Grain-FD 规则具有通用性，不包含任何特定问题的内容。我们在一个试点项目中制定了这些规则；该项目取材于 GrainTraps，以及 46 个 BIRD-FanOut 训练问题。这些问题因 FD 证据会改变模型在其中的结果而被选中（附录D）；GrainTraps 的全部内容和这 46 个问题共同构成设计集。其余 183 个失真训练问题则留出：它们未参与规则设计。这些问题是根据 FD 的结果选出的一组问题的补集，而非随机划分所得。

各项比较衡量的内容。Stats 与无证据的比较，体现了简单统计数据带来的作用；每个 FD 实验组与 Stats 的比较，体现了依赖关系在此基础上带来的作用；每个粒度感知组与提供方式相同的 FD 实验组的比较，则体现了粒度感知带来的作用。在探索性对照中，Stats+Rep 与 Stats 的比较，体现了重复计数带来的作用；Grain-FD 与 Stats+Rep 的比较，大致体现了其余粒度感知证据带来的作用（只是大致如此，因为 Grain-FD 不包含 Stats+Rep 中的逐列计数）。FD 与 FD（工具）在措辞和提供方式上均不同，因此两者分别与 Stats 比较，而非彼此比较。粒度感知组也是唯一明确指出度量列的组，因此其效果也包含了这项标注的影响。

6 结果

模型与运行。 我们测试了八个模型，并根据它们在无证据条件下的 GrainTraps 粒度错误进行分组：低错误 （Claude Son- net 5.5 [6], Claude Haiku 5.5 [4]；2.4%）、中等错误 （DeepSeek V4.1 Flash [19], GPT-5.6 Luna 和 Terra [48], GLM-5.3-Flash [22]；13.2%）以及高错误（Gemini 3.5 Flash-Lite [26], NVIDIA Nemotron 3.5 Super VL 120B-A12B [45]；40.8%）。分组时未使用 BIRD-FanOut，但它给出的顺序相同（无证据条件下的扇出错误分别为 13.8%、17.7%、24.5%；每一步 $p \leq .004$）。这些层级仅用于对结果分组；在 GrainTraps 上，某个层级相对于无证据条件的变化部分源于回归均值，因为层级是在该实验组上确定的。每个模型对每个陷阱问题回答三次，对每个 BIRD-FanOut 问题回答一次；在六个主要组中均进行测试，并且仅在 GrainTraps 和 BIRD 训练集上，在 Stats+Rep 中也进行测试，涵盖这些问题集（表 1 ）。

实验使用最初标注的 BIRD 问题（421 个训练集问题、105 个开发集问题），但 Stats+Rep 除外：它是在标准答案审核后运行的，使用其保留的 384 个训练问题；评分仅基于保留的 486 个问题。总计运行 38,232 次：在 BIRD 上，每个问题、实验组和模型各运行一次；在 GrainTraps 上运行三次。温度采用提供商默认值，智能体步骤最多 50 步。另有三个模型（Gemini 3.8 Flash、Qwen 3.8 27B、GPT-OSS-120B）未能完成所有实验组，因此不纳入统计。

统计检验。 每次证据运行都与其参考组在相同问题、模型和重复实验下的运行配对。我们将每项比较报告为被测试实验组避免的错误（GrainTraps 上为粒度错误，BIRD 上为扇出错误），以运行次数的百分点表示（正值更好）。推断单位是问题。在 Grain- Traps 上，一个基础问题 由一个明确问题及其模糊孪生问题组成；在 BIRD 上，每个问题本身就是一个基础问题。图中给出基础问题上的 95% 自助法区间 [21]；p值来自对每个问题变化进行的符号翻转置换检验，标记为符号检验的情况除外。六项主要检验（Grain-FD 和 Grain-FD（工具）分别与 Stats 比较，涵盖 GrainTraps、BIRD 训练集及其留出问题）在 Holm校正后仍然显著（最大的校正后的p为 $= .012$）。涉及 Stats+Rep 的八项检验属于探索性分析；没有一项通过

表1：问题集根据其审核后的标准答案进行评分（§4）。扇出错误在失真问题上进行测量，这些问题的原始标准查询会产生过度计数；而在 预期的问题上衡量危害，在这些问题中，对重复行进行计数是正确的。GrainTraps 包含 29 个明确问题和 27 个模糊孪生问题。

set	问题
GrainTraps	56 个含陷阱，3 个对照项
BIRD 训练集，失真	229：183 个留作测试，46 个用于设计
BIRD 训练集，预期的	155
BIRD 开发集，失真	32 仍然失真，34 已由 BIRD 修复
BIRD 开发集，预期的	36

该检验族内的 Holm校正（最小的校正后的p为 = .058）。按层级或模型划分的结果仅作描述性分析。

衡量 BIRD 上的扇出。所有运行结果均依据第 §4 节的审核后的标准答案进行评分。由于发现 BIRD-FanOut 问题的审计只能检测重复行，所有失真标准答案都会产生过度计数，因此在 BIRD 上，我们衡量的是扇出错误，而不是所有粒度错误。一个答案如果既过度计数又犯了其他错误，则不属于精确扇出错误。为了捕捉这类答案，一个非目标模型的大语言模型评判器会将其他每个不同的错误答案标记为扇出或非扇出；随后，第二个评判器检查随机样本，并通过审计（§6.3）解决所有分歧。我们报告 BIRD 上扇出错误的三个指标：

- 精确 + 判定（主要指标，称为扇出错误）：精确扇出错误加上被判定为过度计数的答案；
- 仅精确：不使用大语言模型评判器；
- 精确 + 审计标记：不使用大语言模型评判器；采用 Grain-Traps（图10）中的规则。

6.1 谁会产生粒度错误?

每个模型在 GrainTraps 上都会出现粒度错误，而会检查数据的模型出现的错误最少（图 8）。在没有证据的情况下，每个模型的 168 次非对照运行中，粒度错误占比从 2%（Sonnet）到 42%（Nemotron）不等（附录 C）；低错误模型只会在 NULL 划分和扇形陷阱中失败，并能解答每个鸿沟问题。

检查。 检查查询是指模型在作答前运行的、用于测试列之间关系的查询（COUNT DISTINCT、HAVING COUNT、IS NULL），不包括最终提交的查询。两个 Claude 模型在 95–98% 的运行中会进行检查，而且几乎从不出现粒度错误；其他模型的检查频率则低得多。在同一模型内，进行过检查的运行出现的粒度错误更少（Terra 为 7% 对 15%，GLM 为 13% 对 23%，Flash-Lite 为 26% 对 44%），但无论是按模型分别比较，还是仅比较同一问题上的运行，这些差异都不显著（Mantel-Haenszel [41] p > .7）。这表明，检查体现的是模型本身是否谨慎，而不是让模型变得谨慎。在完整模式运行中（§5.2），添加一行提示，告诉模型进行检查
将数据加入较早的提示后，检查率大幅上升（GPT-5.5 为 3% → 21%，Sonnet 为 57% → 94%），但粒度错误并未减少。

6.2 粒度陷阱的证据

在 GrainTraps 上，统计数据有所帮助，函数依赖毫无作用，而感知粒度的函数依赖比统计数据少造成约五个点的粒度错误（图9，左）。GrainTraps 属于设计集，

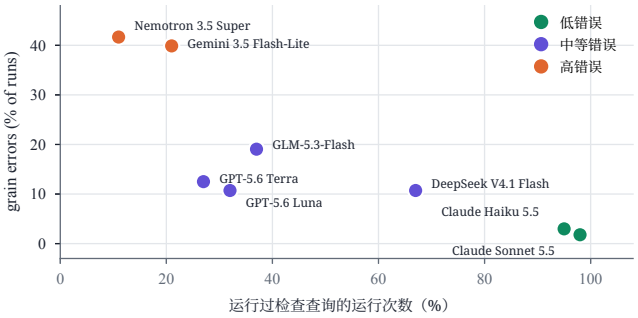


图8：缺少证据时的 GrainTraps：按模型统计，执行检查查询的运行占比及出现粒度错误的运行占比（每个模型各有 168 次非对照运行）；颜色表示等级。检查越多的模型，粒度错误越少：两种 Claude 模型在 95–98% 的运行中都会检查，且几乎从不出现粒度错误。

因此，这些结果表明，证据修正了其针对的失败；留出集测试则在 BIRD 上进行（§6.3）。

图9展示了各实验组之间的配对差异；各实验组的绝对比率列于下文（表2）。

出现粒度错误的运行占比从缺少证据时的 17.4% 降至提供统计数据时的 12.8%（p = .015）。函数依赖和函数依赖（工具）并未带来超出统计数据的额外收益；低错误率模型的表现甚至略有退步，且这些情况全都出现在 NULL 划分中，因为函数依赖会将空值视为一个值。Grain-FD 和 Grain-FD（工具）将比率分别降至 8.3% 和 7.7%，比统计数据低 4.5 和 5.1 个百分点（p = .012 和 .003），准确率也大致同幅提升。大部分收益来自扇形陷阱——这正是度量粒度所针对的失败类型：29 个基础问题中有 5 个贡献了避免出现的 60 个粒度错误中的 35 个。由于收益集中在少数问题上，符号检验（无论变化幅度如何，都对每个基础问题计数一次）的检验力较弱（p = .078 和 .041）。Grain-FD 使六个低错误率和中等错误率模型中的四个不再出现粒度错误；高错误率模型在各自 168 次运行中仍分别出现 39 和 61 个粒度错误（附录 C）。

重复计数还是函数依赖？（探索性分析）仅重复计数（Stats+Rep）就比统计数据少造成 2.0 个百分点的粒度错误（p = .019），达到了 Grain-FD 降幅的 45%，但几乎没有提升准确率。Grain-FD 进一步减少了 2.5 个百分点（不显著），主要体现在扇形陷阱问题上，因为只有它说明了度量粒度（使用 Stats+Rep 时有 49 个粒度错误，而它只有 21 个）；但在鸿沟问题上反而更多（20 个，而不是 10 个）。

6.3 真实数据：BIRD-FanOut

在 BIRD 训练集上，粒度感知函数依赖将扇出错误的运行占比从 18.4% 降至 13.1–13.6%，比使用统计数据少 3.4–3.9 个百分点（p < .001）；在留出问题上以及在不使用任何大语言模型评判器统计错误时，也观察到了这一结果。

在没有证据的情况下，对229个失真训练集问题的运行中，有12.9%准确返回了失真标准答案的结果。若将那些虽然过度计数但同时犯有另一处错误的答案也计算在内，则18.4%的运行属于扇出错误（表2）。与统计数据相比，Grain-FD可避免3.4个百分点的错误，Grain-FD（工具）可避免3.9个百分点（两者均为 p < .001；图9，右），普通函数依赖可避免1.7和2.3个百分点，而仅使用统计数据相对于无证据只能避免1.4个百分点（不显著）。与采用相同提供方式的普通函数依赖相比，具备粒度感知能力的附加机制进一步避免了

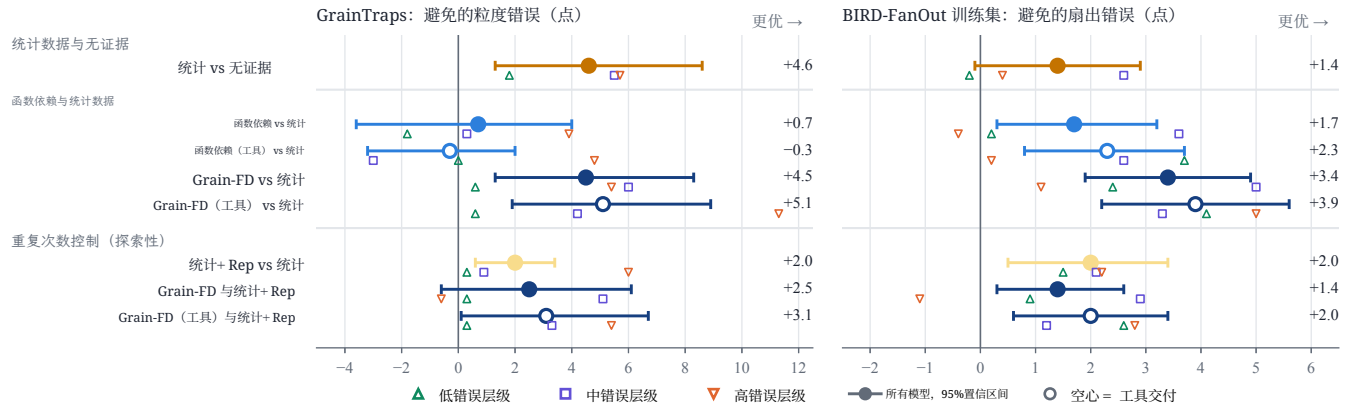


图9：与各自参照组相比，受测实验组避免的错误，以运行次数的百分点表示（越靠右越好）：GrainTraps 上的粒度错误（对 Grain-FD 组而言为样本内数据），以及 BIRD-FanOut 训练集上的扇出错误（精确加判定；229 个问题）。大圆点：全部八种模型，按问题计算的 95% 自助法置信区间（一个陷阱问题及其模糊孪生问题计为一个）；小标记：层级（描述性）；空心标记：工具交付。在 GrainTraps 上，FD 相较统计数据没有带来提升；在 BIRD 上则提升了 1.7–2.3 个百分点；Grain-FD 组带来的提升最大。完整表格（含准确率）见附录 E。

表2：各实验组的错误率，以运行次数的百分比表示：GrainTraps上的粒度错误（每个实验组运行1,344次）、229个失真BIRD-FanOut训练集问题上的扇出错误（1,832次）以及183个留出集问题上的扇出错误（1,464次），还有229个问题上的准确率。粗体表示每列中的最佳值；Stats+Rep为探索性结果。

arm	GrainTraps	BIRD 训练集	留出集	BIRD 准确率
无证据	17.4	18.4	16.1	60.3
统计	12.8	17.0	15.1	61.2
统计+重复	10.8	15.1	13.9	62.3
FD	12.1	15.3	13.9	62.0
FD（工具）	13.1	14.7	12.9	62.0
Grain-FD	8.3	13.6	12.4	**63.3**
Grain-FD（工具）	**7.7**	**13.1**	**12.2**	63.2

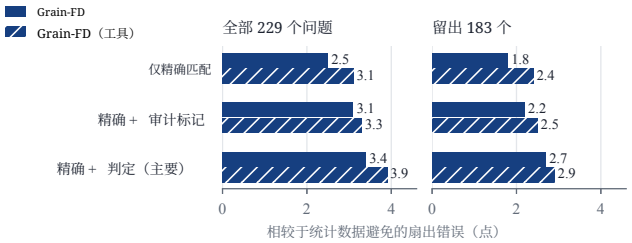


图10：BIRD-FanOut训练集：在三项指标下，相较于统计所避免的扇出错误，其中两项指标不使用大语言模型评审员。左：全部229个问题；右：183个留出问题。斜线填充：工具交付。

两种交付方式中的差异均为 1.6 点（[0.3、3.1] 和 [0.2、3.1]；p = .03 和 .04，未纳入六项主要检验）；在留出问题上，差异分别为 1.6 点（p =. 05）和 0.7 点（不显著）。

留出问题、模型和数据库。在 183 个留出问题上，Grain-FD实验组比统计数据实验组少出现 2.7 和 2.9 个扇出错误（两者均 p ≤.001）；在依据 FD 的结果选出的 46 个设计问题上，效果更大（6.0 和 8.2 点）。在 8 个模型中，有 7 个模型的扇出错误少于统计数据实验组（另有 1 个持平）；与 Grain-FD（工具）相比，8 个模型的扇出错误均更少。将重采样单位从问题改为 46 个数据库后，两种效果仍然存在（区间为 [1 6, . . 和 . . 点）。5 3] [1 9, 6 1]

与无证据相比，Luna、Sonnet 和 Terra 的扇出错误降幅最大，而 Nemotron 和 Haiku 几乎没有变化（图11）；Haiku 的运行轨迹表明，它能察觉重复，并遵循提示中的行级公式（§6.4）。每个问题仅运行一次，因此各模型之间的差异仅作描述性参考。

无评审指标与评审员的可靠性。 两项无需评审员参与的指标得出了相同结论（图 10）：仅统计精确扇出错误，以及统计精确扇出错误再加上审计标记为确定扇出的其他答案。评审员发现了精确匹配未能识别的过度计数，但并未造成这一现象：在错误答案中，那些并非精确扇出错误的答案里，评审员判定为扇出的比例随证据增加而下降（无证据时为 20.6%，有 Grain-函数依赖时为 14.4%），审计标记的比例也同样下降（11.8%、7.1%）。若将评审员判定为在错误粒度上分组的少数答案（约占运行次数的 1%）也计为粒度错误，则结果相同（比统计数据高 3.1 和 3.7 个百分点，在留出问题上高 2.4 和 2.7 个百分点，p ≤.007）。

第一位评审员 Gemini 3.6 Flash 会看到问题和提示、模式、两份标准答案及其结果，以及附带结果的答案，但看不到审计标记。Claude Opus 5.5 在随机抽样中有 98% 的判断与其一致（Cohen's κ [16] 0.91），并对 Gemini 与审计意见不一致的所有答案作出裁定。五个可以查询各个数据库、但看不到标签的 Opus代理，在 50 个答案中有 47 个与其一致（κ 0.88）；少数分歧表明，经评审判定的扇出结果或许略有低估。这些检查并非相互独立，因为两位评审员看到的是同一提示，其中说明该问题因过度计数而被标记；此外，Opus 既负责裁定分歧，也负责运行这些智能体。标准答案审核（§4）在没有证据的情况下将准确率提高了 11 点，但对这些结果的影响

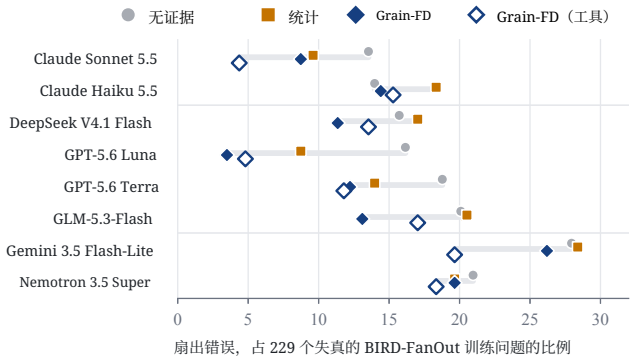


图11：BIRD-FanOut 训练集上各模型的扇出错误数（每个问题和实验组各运行一次）。不同层级以行分隔：低、中、高错误。与无证据相比，Grain-FD实验组在 Luna、Sonnet 和 Terra 上减少的错误最多；Haiku 和 Nemotron 的变化微乎其微。

扇出结果最多为 0.5 点。附录D列出了完整的检查。

重复计数还是函数依赖？（探索性分析）仅重复计数就贡献了约一半的错误减少：Stats+Rep 比仅使用统计数据多减少 2.0 点（$p = .013$），相当于 Grain-FD 错误减少量的 58%（95%区间为 21–91%）。其余粒度感知证据又减少了 1.4 点；使用工具时则多减少 2.0 点，在留出问题上也有类似效果（分别为 1.5 和 1.6 点，$p = .028$ 和 .036）；普通函数依赖在重复计数之外没有额外贡献。工具带来的增量更为稳健，因为仅看精确扇出错误时仍然成立，且在 8 个模型中的 7 个上都成立。这个探索性检验系列中，没有任何检验在 Holm 校正后仍显著。

准确率与开发集。准确率低估了这些影响，因为这些问题中的大多数错误并非扇出错误，而且避免一个扇出错误往往会转化为另一个错误（§6.4）。与无证据相比，Grain-FD 和 Grain-FD（工具）将准确率从 60.3% 提高了 3.0 和 2.8 个百分点（$p = .003$ 和 .009）；在留出问题上，准确率提高了 1.8 个百分点，但不显著。与统计数据相比，两者将准确率提高了约 2 个百分点（$p = .031$ 和 .07），主要体现在低错误和中错误层级；在留出问题上提高了 0.6 个百分点（不显著），因此我们不声称函数依赖比统计数据带来了准确率提升。没有任何实验组能使高错误率模型的准确率提高超过 1.5 个百分点，不过，与无证据相比，Grain-FD（工具）将这些模型的扇出错误减少了 5.5 个百分点。在开发集上，清理后的发布版本中仍有失真的 32 个问题才可能出现扇出错误，数量太少，无法进行检验（扇出错误：无证据组在 256 次运行中有 38 次，Grain-FD（工具）组有 25 次，$p = .30$）；开发集上的准确率比较均无显著差异。在 85 个复现旧版标准答案结果的开发集答案中，只有 13 个几乎逐字照搬了其 SQL，因此记忆公开的旧版标准答案所起的作用很小。

6.4 为什么证据在真实数据上的作用较小？

Grain-FD 将 GrainTraps 上的粒度错误减少了一半，但在 BIRD 上仅将扇出错误减少了四分之一，这主要是因为问题本身，以及部分数据几乎没有重复。 从无证据到 Grain-FD，GrainTraps 上的粒度错误从运行次数的 17.4% 降至 8.3%；在该数据集中，几乎每个错误都是粒度错误；而 BIRD 训练集上的扇出错误则从 18.4% 降至 13.6%。图12将 BIRD 的 229 个失真问题按照数据和问题的属性进行划分。

重复、规模与噪声。当过度计数幅度较小时，证据几乎无法发出警告。对于 17% 带有陷阱查询的 BIRD 问题，过度计数导致的答案变化不到 10%（在 GrainTraps 中，这一比例为 4%）。这些问题中，重复的行很少，因此证据显示每个值对应的行数接近一行；但它们仍频繁出现扇出错误，而 Grain-FD 仅能消除其中 19%，相比之下，过度计数幅度较大时可消除 30%。数据的其他特性影响不大。BIRD 的表比陷阱表更大、更杂乱（行数中位数为 3,118，而陷阱表为 70；三分之一的表中有空值；30% 的表中存在只对除少数行外的所有行都成立的依赖关系，而陷阱表中从未出现这种情况），然而无论是否具有这些特性，Grain-FD 都能将扇出错误减少 24–27%。

这些问题。BIRD 的问题与 GrainTraps 的问题有两点差异，而这两点更为重要。

1. **提示给出了失真公式**。BIRD 的大多数问题都附带提示（BIRD 的“evidence”字段），由其标注人员根据原始标准查询编写。229 个失真训练问题中，有 127 个的提示写明了标准答案中的公式，例如“每辆车的平均价格 = 平均值(price) where 国家 = ‘Japan’”。有了这样的提示，扇出错误的发生频率反而翻倍（占运行次数的 24.1%，而非 11.3%），Grain-FD 将其减少了 20%，而非 40%。GrainTraps 没有提示。一名 Claude 智能体阅读了 40 次仍然出现精确扇出错误的 Grain-FD 运行记录，每个模型 5 次。其中有 21 次，模型看到了重复项，却仍然选择了行级解读；这 21 次中有 15 次带有公式提示（GLM：“提示明确写着 avg(price).....我就采用连接查询答案吧”）。

2. **少数棘手、有时也存在争议的问题造成了大量错误**。在没有证据时，229 个问题中有 22 个造成了 45% 的扇出错误；使用 Grain-FD 时，这些问题造成了 52% 的扇出错误。而在 GrainTraps 上，没有任何问题在大多数 Grain-FD 运行中保留粒度错误。在这 40 次运行中，有 11 次里，问题本身或其修正后的标准查询都允许另一种解读。

准确率。第三个差异解释了为何准确率提升幅度较小，但无法解释为何避免的扇出错误较少。在 BIRD 上，21% 的运行因其他原因失败（GrainTraps 上为 3%），因此避免一种错误后，往往会变成另一种错误。在没有证据时陷入陷阱的运行中（在 BIRD 上是精确的扇出错误，在 GrainTraps 上是粒度错误），Grain-FD 在 GrainTraps 上将 58% 的运行转化为正确答案，但在 BIRD 上仅转化了 36%。

证据本身。在 40 次运行中，有 13 次证据不足。所需事实可能缺失（某一列被标注为维度而非标识符，因此其重复计数信息丢失）、被埋在 100 多行证据中，或虽有说明却未被使用。

6.5 何时应计数重复行

表明行会重复的证据，可能会促使模型对那些问题的行进行去重，而这些问题本来需要计数。我们在应该对重复行进行计数的场景中对此进行测试：在 GrainTraps 的三个控制问题上，每个模型在每次运行和每个实验组中都能答对；以及在 BIRD-FanOut 的 191 个预期问题上（开发集和训练集）。在提示中提供证据后，没有模型在预期问题上的准确率显著低于不提供证据时（最大降幅为 1.6 点）。在工具组中，有两个模型确实会在不该去重的地方进行去重

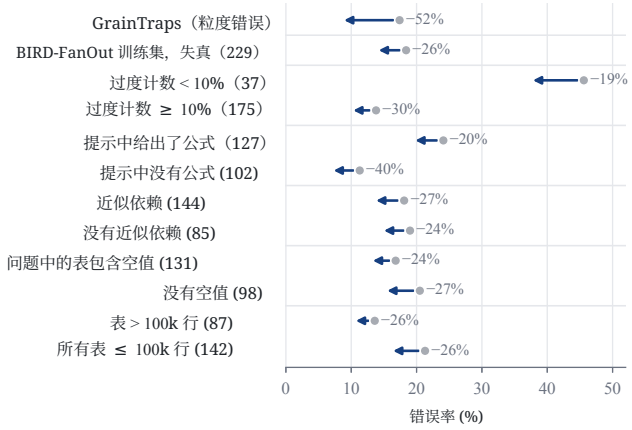


图12：按问题分组（问题数量）统计的扇出错误率：无证据（圆点）与使用 Grain-FD（箭头）时的对比；上排：GrainTraps 上的粒度错误。箭头长度表示错误率下降的百分点；标签表示相对降幅，即下降幅度占无证据时错误率的比例（18.4% → 13.6% 是 −26%）。过度计数组涵盖了带有陷阱查询的 212 个问题，这是衡量过度计数所必需的。过度计数规模和提示公式很重要；空值、近似依赖和表大小则无关紧要。

。与统计组相比，Nemotron 3.5 Super 使用 Grain-FD（工具）时准确率下降 10.5 点，GPT-5.6 Luna 下降 5.3 点；其他模型则没有表现出一致的下降。与无证据相比，这些降幅较小且不显著，而且它们来自逐模型比较，而非六项主要检验（§6），因此表明工具交付可能存在风险，但尚不足以证明已经造成危害。使用 FD（工具）时，Nemotron 下降 7.9 点，Luna 下降 5.3 点。对于高错误层级，与统计数据相比，降幅分别为 4.7 点（p = .037）和 4.2 点（p = .058），经修正后不再显著；与无证据相比，降幅分别为 −2.1 点和 −1.6 点；在标准答案审核之前，第一项为 −4.2，因此其中一部分属于金标准错误。

6.6 剩余错误

在完整模式运行中，GrainTraps 答案中大多数被评为“其他”（§5.1）的答案，仍然是粒度错误或对表之间关系的其他误读，我们将其称为数据模型错误。两位评审员 Claude Opus 5.5（在这些运行中也是目标模型）和 GPT-5.6 Terra 对每个不同的“其他”答案进行分类（359 次运行中的 327 个不同查询），并查看问题、陷阱说明及解答、模式和两份结果。他们对 81% 的答案在原因判断上意见一致（κ = 0.72），对其属于粒度错误还是数据模型错误的判断有 86% 一致。图13展示了 Opus 判断器判定的各类原因。

两位评审员都将 327 个答案中的 210 个判定为粒度或数据模型错误（仅 Opus 判定为 229 个），因此只统计粒度错误会低估模型误读数据模型的频率。粒度错误的答案中，大多数（166 个中有 158 个）属于先排序后关联问题，措辞含糊，留下了一些解释空间。Opus 5.5 给出的五个“其他”目标模型答案中，有四个在没有明细键的情况下对明细行应用 DISTINCT，因此合并了实际上彼此不同的行，这与扇出恰好相反。对获得证据的开放模型运行轨迹进行分析，发现证据未被采用有三个原因

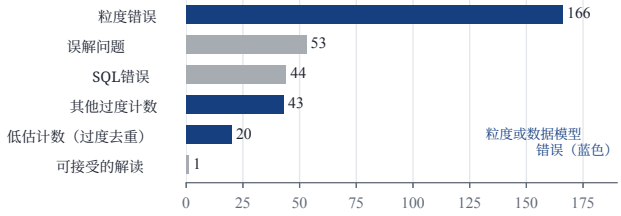


图13：327 个不同“其他”陷阱答案的原因（Opus 判断器；完整模式运行）。蓝色：Opus 判断器计为粒度或数据模型错误的原因。

：逐表事实未通过连接进行合并；仅限少数列的工具调用遗漏了携带度量粒度的那一列；先入为主的判断凌驾于证据之上（“假设每个产品只有一个部门”，这句话紧接在工具指出一个产品可以有多个部门之后）。

6.7 将审计作为离线验证器

审计还可以在任何人使用模型答案之前对其进行检查。在完整模式运行中，我们对每个不同的已提交答案都运行了审计（BIRD-FanOut 开发集上有 2,161 个，在 150 个普通 Mini-开发问题上有 1,807 个）。它能捕获大多数过度计数问题：在复现失真标准答案的答案中，有 70–75% 会被标记（其中 12–26% 收到明确标记，其余收到有歧义或重复值标记）。它很少标记正确答案：在带有扇出的问题中，开发集问题的比例为 1%；在普通问题中为 5%，其中仅 0.3% 收到明确标记。不过，它无法判断意图。在预期问题中，它会标记 53% 的正确答案，因为标记仅表示行存在重复，并不表明问题是否希望将它们只计一次；这一判断必须交由模型或用户作出。

6.8 成本

工具组每次运行大约会增加两次工具调用。证据组读取的输入词元最多是无证据组的两倍。BIRD 训练集每次运行的中位数分别为：无证据时 4.4k 个词元，使用 Stats 时 4.9k 个词元，使用 Stats+Rep 时 5.4k 个词元，使用 FD 和 FD（工具）时 7.8k 个词元，使用 Grain-FD 时 8.7k 个词元，使用 Grain-FD（工具）时 7.7k 个词元。Grain-FD 的内容最长，因为它增加了重复计数、感知 NULL 的函数依赖和度量粒度；对于某些 BIRD 表，内容超过 100 行。所有证据都按表计算一次并缓存；在 BIRD 训练集最大的数据库（最大 5 GB）上，首次生成某个表的 Grain-FD 配置文件最多需要 27 分钟。

7 讨论

教训一：标准查询需要以数据为依据，对整个查询进行审查。 当 BIRD 的标准答案失真时，避开陷阱的模型反而会被判错。审查还必须涵盖提示：BIRD 的标注人员是针对失真的标准答案编写这些提示的，而它们往往会指示过度计数（229 个失真训练问题中，有 127 个附带公式提示）；模型若发现了重复，并遵循此类提示，便是在化解两条指令之间的冲突，而非忽视证据。仅修正扇出仍使我们修正后的训练集标准答案中每五个就有一个留有另一处 BIRD 错误；修正这些错误后，同一批问题的测得准确率提高了 11 个百分点（§6.3）。

审查本身需要用到数据，因为两位审查者可能会一致同意一个返回空结果的修正方案。

教训 2：证据应说明粒度。逐列计数已经能揭示键、重复情况（图 7 中，1,200 个订单对应 4,185 行）以及缺失项。在 GrainTraps 上，正确的函数依赖标签（“可能有多个”）并未提供额外信息；在我们的探索性对照中，它们在 BIRD 上也没有超出重复计数所提供的信息。真正有帮助的是能提供粒度条件（§2.2）所需输入的证据：每个度量的粒度、连接会重复多少行，以及空值会在何处破坏依赖关系。MCI-SQL 中依赖关系标签带来的微小影响 [60], 是根据准确率衡量的，与我们的准确率结果一致；直接衡量扇出错误则能更清楚地显示其影响。

教训 3：应直接衡量扇出错误。在真实问题中，大多数错误并非扇出错误，而且过度计数也可能在其他方面存在差异，因此准确率和完全匹配率会低估证据的作用。逐一判断每个错误答案是否属于扇出错误是可行的，因为大语言模型评审器、审计和数据探查智能体的判断高度一致（§6.3），尽管这些判断并非相互独立。

未来工作：关于模型自身查询的事实。逐表事实让模型自行跨连接组合这些信息，但它往往做不到；而将查询与数据进行检查，则能发现大多数精确扇出错误（§6.7）。下一步可以针对模型查询所用的每个关系，报告连接如何重复行，以及每个聚合度量的粒度；这些信息根据逐表依赖关系通过规则推断，只有规则无法确定时才从数据中计算。与需要有人声明目标模型的本体和语义层不同，这些事实可以直接从数据中得出。

局限性。

- 范围。 我们审计了一个基准测试系列（BIRD 开发集、训练集和 Mini-Dev），因此其他基准测试的标准答案中包含多少扇出，仍是未知的。GrainTraps 规模较小，是由 Claude 智能体生成的合成数据集，仅保留强模型能够解答的问题，且没有留出划分。没有证据表明两个 Claude 模型在其中的粒度错误最少（168 次运行中分别为 3 次和 5 次，而其他模型为 18 至 70 次），因此跨模型比较可能对它们有利；实验组的影响是在每个模型内部测量的。我们只使用一种方言（SQLite；审计使用 rowid）、每个层级使用两到四个模型，并且每道 BIRD 问题只运行一次（在 GrainTraps 上，六个主要组中三次运行的单元格有 14.5% 的结果不一致）；此外，我们只向模型展示参考查询读取的表，这使任务比实际情况更容易。- 设计。 Grain-FD 规则是在设计集上制定的（GrainTraps，以及根据 FD 的结果选出的 46 道 BIRD 训练集问题）。这些规则具有通用性，不包含针对特定问题的内容；183 道留出问题用于检验这些规则，但陷阱测试结果是在样本内得出的，此后没有编写新的陷阱问题，而且四个要素也没有逐一进行消融。度量粒度是单列，因此，对于按列对重复的度量（在检查的 650 个 BIRD 度量中有 6 个），报告为随行变化。各实验组在 2026 年 11 月的不同日期运行，使用相同代码；Stats+Rep 最后运行，其他组没有在同一天重复运行，因此提供商侧模型更新可能会改变组间比较，尤其是重复计数与函数依赖之间的划分。 Stats+Rep 未在开发集上运行，而且它的文本并非 Grain-FD 的组成部分，因此两者之间的差异不能算作严格的消融实验。FD 与 FD（工具）在措辞和提供方式上都有差异，而且各实验组的长度也不同（§6.8）。
- 标签。没有任何人员对数据进行标注或检查。列角色和 BIRD 标签由大语言模型评审员给出，仅 Claude 智能体对抽样数据进行了检查（其中一个角色错误导致遗漏了一条必要事实）。BIRD 的扇出错误判断依赖于大语言模型评审员；这些评审员彼此之间、与审计结果以及数据探查智能体（与第二位评审员使用相同模型）的判断都高度一致。指标是在运行结束后才选定的，当时标准答案审核揭示了精确匹配会漏掉多少过度计数；无需评审员的指标也得出了相同结论（图10）。标准答案审核本身也依赖于大语言模型评审员，其中两位同时还是目标模型（第一阶段使用 Claude Haiku 5.5，第二阶段使用 GPT-5.6 Terra）。一位 Claude 智能体检查了促使审核进行的 21 条标准答案，以及 5 项获采纳的修正（4 项显然正确，1 项有充分依据）；每项未返回答案的修正也都经过了数据检查。标准答案审核未涵盖 BIRD-FanOut 中的 40 个问题；此外，17 个失真问题没有陷阱查询，因此可以判定其扇出错误，却无法进行精确匹配。运行时向模型展示了原始标准查询读取过的表；任何固定标准查询都不会读取其他表。§6.4 的轨迹分析属于定性分析，是一位 Claude 智能体对 40 次运行进行的一轮编码分析。

8 结论

在文本转 SQL 中，扇出错误往往隐蔽且常见，并且受一个条件支配：当关系中每个实体在其度量粒度上仅出现一次时，重复敏感聚合才是正确的。该条件所需的输入——度量粒度以及连接所引起的多重性——是模式无法展示的数据属性。在数据上进行测试后，该条件揭示出 BIRD 的问题中有少数几个百分点的失真标准答案。将其作为具有粒度意识的函数依赖提供给大语言模型智能体后，在真实问题上，经大语言模型判定的扇出错误比例从没有证据时运行结果的 18.4%、有统计数据时的 17.0%，降至 13.1–13.6%；在留出问题上，则从 16.1% 和 15.1% 降至 12.2–12.4%。准确率仅上升约 3 个百分点，因为剩余的主要障碍是其他错误、要求采用失真公式的提示，以及数据中不明显的过度计数。正如审计所做的那样，针对数据检查模型自身的查询，是自然的下一步。BIRD-FanOut、GrainTraps、审计工具和证据代码可在 https://github.com/ahmed-mahran/misqal 获取。

参考文献

[1] Ziawasch Abedjan、Lukasz Golab 和 Felix Naumann。2015年。《关系数据剖析：综述》。*The VLDB Journal* 24，4（2015年），557–581。https://doi.org/10.1007/ s00778-015-0389-y [2] Ziawasch Abedjan 和 Felix Naumann。2011年。《推进唯一列组合的发现》。载于第*20*届 *ACM* 信息与知识管理国际会议（*CIKM*）论文集。ACM，1565–1570。https://doi.org/ 10 1145 2063576 2063801 . / 。Dean Allemang 和 Juan Sequeda。[3] 2024年。《借助本体提高大语言模型问答系统的准确率》。载于《语义网——*ISWC 2024*：第*23*届国际语义网会议论文集，第三部分（计算机科学讲义），第15233卷。Springer，324–339。https://doi.org/10.1007/978- 3-031-77847-6_18 arXiv:2405.11706 预印本：《提高问答的大语言模型准确率：本体来助阵！》，arXiv:2405.11706。[4] Anthropic。2026年。《系统卡：Claude Haiku 5.5》。https://www.anthropic.com/ claude-haiku-5-5-system-card。日期为2026年10月7日。[5] Anthropic。2026年。《系统卡：Claude Opus 5.5》。https://www.anthropic.com/ claude-opus-5-5-system-card。日期为2026年9月22日。

[6] Anthropic。2026年。《系统卡：Claude Sonnet 5.5》。https://www.anthropic.com/ claude-sonnet-5-5-system-card。日期为2026年9月28日。[7] Siegfried Bell 和 Peter Brockhausen。1995年。《关系数据库中的数据依赖关系发现》。LS-8 报告 14。多特蒙德大学，计算机科学系，第八教席（人工智能），多特蒙德，德国。https: //d-nb.info/1103028413/34[8] Laure Berti-Équille、Hazar Harmouch、Felix Naumann、Noël Novelli 和 Sar- avanan Thirumuruganathan。2018年。《从含缺失值的关系数据中发现真正的函数依赖关系》。VLDB Endowment 论文集 11, 8 2018 880 892 10 14778 3204028 3204032 ( )， – 。[9] https://doi.org/ 。/ 。Adithya Bhaskar、Tushar Tomar、Ashutosh Sathe 和 Sunita Sarawagi。2023年。《在歧义情形下评测并改进文本到 SQL 生成》。载于《2023年自然语言处理实证方法会议论文集》（EMNLP）。计算语言学协会，7053–7074。https: // doi.org/ 10 18653 1 2023 436 . /v / .emnlp-main。BIRD 团队。 。[10] 2025 BIRD-SQL 开发集，清理后的发布版本 2025年11月6日（bird_sql_dev_20251106）。Hugging Face 数据集，https://huggingface.co/ datasets/birdsql/bird_sql_dev_20251106。[11] BIRD 团队。2026年。BIRD-SQL Mini-Dev（2026 年更新版）。GitHub 仓库 https:// github.com/bird-bench/mini_dev 和 Hugging Face 数据集 https://huggingface. co/datasets/birdsql/bird_mini_dev。包含 BIRD 开发集中的 500 个问题，首次发布于 2024年；2026年更新版发布于2026年10月28日。[12] Yahia Bsat。2025年。《Snowflake Cortex Analyst：防止连接幻觉和重复计数》。Snowflake 工程博客，https://www.snowflake.com/en/blog/engineering/snowflake-cortex-analyst- introducting-joins-complex-schemas/。发布于2025年2月20日。[13] Diego Calvanese、Benjamin Cogrel、Sarah Komla-Ebri、Roman Kontchakov、Da- vide Lanti、Martin Rezk、Mariano Rodriguez-Muro 和 Guohui Xiao。2017年。《Ontop：在关系数据库上回答 SPARQL 查询》。语义网 8，3（2017年）， 471 487 10 3233 160217 – 。https://doi.org/ 。/SW- Surajit [14] 1994 Chaudhuri 和 Kyuseok Shim。 。《在查询优化中纳入分组》。载于《第20届超大型数据库国际会议论文集》（VLDB）。智利圣地亚哥，354–366。https://www.vldb.org/conf/1994/ P354.PDF[15] Rohit Chaurasia。2026年。《 grainguard：静态检测 SQL 和 dbt 中的扇出聚合》。GitHub 仓库，版本 0.1.0, https://github.com/rohitchaurasia195- web/GRAINGUARD。[16] Jacob Cohen。1960年。《名义量表的一致性系数》。教育与心理测量 20，1（1960年），37–46。https://doi.org/10.1177/ 001316446002000104[17] Thomas Connolly 和 Carolyn Begg。2015年。《数据库系统：设计、实现与管理的实用方法》（第6版）。Pearson，哈洛。[18] dbt Labs。2026年。《连接》。dbt 开发者中心，MetricFlow / dbt 语义层文档，https://docs.getdbt.com/docs/build/join-logic。[19] DeepSeek-AI。2026年。《 DeepSeek-V4.1-Flash：突破 KV 缓存压缩的极限》。技术报告 arXiv:2609.19969。DeepSeek-AI。arXiv:2609.19969 [计算机科学：自然语言处理] https://arxiv.org/abs/2609.19969[20] Minghang Deng、Ashwin Ramachandran、Canwen Xu、Lanxiang Hu、Zhewei Yao、Anupam Datta 和 Hao Zhang。2025年。《ReFoRCE：具备自我改进、共识约束和列探索能力的文本到 SQL 智能体》。arXiv: 2502 00675 [ ] 2502 00675 . 计算机科学：自然语言处理 [21] 1993 https://arxiv.org/abs/ 。Bradley Efron 和 Robert J. Tibshirani。 。《自助法导论》。Chapman & Hall，纽约。https://doi.org/ 10 1201 9780429246593 . / GLM- -团队。 。《 [22] 5 2026 5 GLM- ：从氛围编程到智能体工程》。技术报告 arXiv:2602.15763。Z.ai。arXiv:2602. 15763 [计算机科学：机器学习] https://arxiv.org/abs/ 2602.15763 GLM-5 系列报告，据 GLM-5.3-Flash 模型卡要求引用（https://huggingface.co/zai-org/GLM-5.3-Flash）。[23] Google Cloud。2026年。《关系（用于连接）》。Looker 文档，LookML 参考，https://docs.cloud.google.com/looker/docs/reference/param-explore- j oin-relationship。声明每个连接的关系为一_对_一、多_对_一（默认）、一_对_多或多_对_多。[24] Google Cloud。2026年。《理解对称聚合》。Looker 文档，https://docs.cloud.google.com/looker/docs/best-practices/understanding-symmetric-aggregates。[25] Google DeepMind。2026年。《Gemini 3.1 Pro 模型卡》。https://deepmind.google/ models/model-cards/gemini-3-1-pro/。发布于2026年2月19日。[26] Google DeepMind。2026年。《Gemini 3.5 Flash-Lite 模型卡》。https://deep mind. google/models/model-cards/gemini-3-5-flash-lite/。发布于2026年7月21日。[27] Google DeepMind。2026年。《Gemini 3.6 Flash 模型卡》。https://deepmind.google/ models/model-cards/gemini-3-6-flash/。发布于2026年7月21日。[28] Omer Hochman。2026年。《你的文本到 SQL 模型并没有基准测试所说的那么差，真正有问题的是标准 SQL》。DEV Community 博客文章，https://dev.to/omer_ hochman/your-text-to-sql-model-isnt-as-wrong-as-your-benchmark-says-the-gold-sql-is-p16。发布于2026年8月7日。[29] Stephan Hoffmann。2022年。《SQL 重复计数消解器》。美国专利 11,416,488 B2, https://patents.google.com/patent/US11416488B2/en。申请号 US16/804,946，申请于 2020年2月28日，授权于2022年8月16日；专利权人为 insightsoftware.com I nternational Unlimited。[30] David Howe。2001。《数据库设计的数据分析》（第3版）。Butterworth-Heinemann，牛津。[31] Ykä Huhtala、Juha Kärkkäinen、Pasi Porkka 和 Hannu Toivonen。1999。TANE：一种用于发现函数依赖和近似依赖的高效算法。《计算机杂志》 42, 2 1999 100 111 10 1093 42 2 100 （），–。[32] 2026 https://doi.org/。/comjnl/。。Tengjun Jin、Yoojin Choi、Yuxuan Zhu 和 Daniel Kang。Pervasive Annotation Errors Break Text-to-SQL Benchmarks and Leaderboards。《 VLDB Endowment 论文集》19，5（2026年），931–944。https://doi.org/10.14778/3796195.3796206 arXiv:2601.08778 短版：“文本到 SQL 基准测试已失效：标注错误深度分析”，CIDR 2026。[33] Nikos Karayannidis。2026。粒度理论：数据管道中的类型级粒度正确性。arXiv:2601.00995 [cs.DB] https://arxiv.org/abs/2601.00995 v3，2026年10月2日。[34] Ralph Kimball 和 Margy Ross。2013。《数据仓库工具包：维度建模权威指南》（第3版）。Wiley，印第安纳波利斯，印第安纳州。[35] Anthony C. Klug。1980。计算关系表达式上的约束。《ACM 数据库系统汇刊》5，3（1980年9月），260–290。https://doi.org/10.1145/320613.320615[36] Henning Köhler 和 Sebastian Link。2016。SQL 模式设计：基础、范式与规范化。载于《2016年数据管理国际会议论文集》（SIGMOD）。ACM，267–279。https://doi.org/ 10 1145 2882903 2915239 。/。Jan Kossmann、Daniel Lindner、Felix [37] Naumann 和 Thorsten Papenbrock。2022。面向查询优化的工作负载驱动型惰性数据依赖发现。载于第12届创新数据系统研究会议（CIDR）。美国加利福尼亚州查米纳德。https://www.vldb.org/cidrdb/papers/2022/p70-kossmann.pdf[38] Fangyu Lei、Jixuan Chen、Yuxiao Ye、Ruisheng Cao、Dongchan Shin、Hongjin Su、Zhaoqing Suo、Hongcheng Gao、Wenjing Hu、Pengcheng Yin、Victor Zhong、Caiming Xiong、Ruoxi Sun、Qian Liu、Sida Wang 和 Tao Yu。2025。Spider 2.0：评估语言模型在真实企业文本转 SQL 工作流中的表现。载于第十三届国际学习表征会议（ICLR）。arXiv:2411.07763 https://proceedings.iclr.cc/paper_ files/paper/2025/hash/46c10f6c8ea5aa6f267bcdabcb123f97-Abstract-Conference.html[39] Hans-Joachim Lenz 和 Arie Shoshani。1997。联机分析处理和统计数据库中的可汇总性。载于第九届科学与统计数据库管理国际会议论文集（ SSDBM）。IEEE 计算机学会，132–143。https://doi.org/10.1109/SSDM.1997.621175[40] Jinyang Li、Binyuan Hui、Ge Qu、Jiaxi Yang、Binhua Li、Bowen Li、Bailin Wang、Bowen Qin、Ruiying Geng、Nan Huo、Xuanhe Zhou、Chenhao Ma、Guoliang Li、Kevin C. C. Chang、Fei Huang、Reynold Cheng 和 Yongbin Li。2023。LLM 能否已胜任数据库接口？一个面向大规模数据库落地文本转 SQL 的大型基准测试。载于《神经信息处理系统进展》第36卷（NeurIPS 2023），数据集与基准测试专题。42330–42357。https://doi.org/10.52202/075280-1835 arXiv:2305.03111[41] Nathan Mantel 和 William Haenszel。1959。疾病回顾性研究数据分析的统计学方面。《美国国家癌症研究所杂志》 22, 4 1959 719 748 10 1093 22 4 719 （），–。[42] 2026 https://doi.org/。/jnci/。。Toby Mao。SQLGlot：Python SQL 解析器和转译器。GitHub 仓库，https://github.com/tobymao/sqlglot。使用版本 30.27.0。[43] Jose-Norberto Mazón、Jens Lechtenbörger 和 Juan Trujillo。2009。多维建模中可汇总性问题综述。《数据与知识工程》 68, 12 2009 1452 1469 10 1016 2009 07 010 （），–。[44] 2026 https://doi.org/。/j.datak/。。英伟达。Auto Ontology。GitHub 仓库，https://github.com/NVIDIA/auto-ontology。配套仓库中提供 BIRD 评测工具：https://github.com/NVIDIA/ontology-sql-eval。[45] 英伟达。2026。Nemotron 3.5 Super 和 Lightning：面向智能体人工智能的开放、高效混合专家混合 Mamba-Transformer 模型。技术报告 arXiv:2610.38095。英伟达。arXiv:2610.38095 [cs.CL] https://arxiv.org/abs/2610.38095[46] OpenAI。2025。gpt-oss-120b 和 gpt-oss-20b 模型卡。模型卡 arXiv:2508.10925。OpenAI。arXiv:2508.10925 [cs.CL] https://arxiv.org/abs/2508.10925[47] OpenAI。2026。GPT-5.5 系统卡。https://deploymentsafety.openai.com/gpt-5-5。发布于2026年4月23日；更新于2026年4月24日和8月19日。[48] OpenAI。2026。GPT-5.6 系统卡。https://deploymentsafety.openai.com/gpt-5-6。涵盖 GPT-5.6 Sol、Terra 和 Luna。发布于2026年7月9日；更新于2026年8月3日和19日。[49] Thorsten Papenbrock 和 Felix Naumann。2016。发现函数依赖的混合方法。载于《2016年数据管理国际会议论文集》（SIGMOD）。ACM，821–833。https://doi.org/10.1145/2882903.2915203[50] Glenn Norman Paulley。2000。在查询优化中利用函数依赖。博士论文。滑铁卢大学，滑铁卢，安大略省，加拿大。https://cs.uwaterloo.ca/research/tr/2000/11/CS-2000-11.thesis.pdf 另见技术报告 CS-2000-11，滑铁卢大学计算机科学系。[51] Aravind Sasidharan Pillai。2026。GROUND：通过受治理的语义定义减少基于大语言模型的企业分析中的幻觉。arXiv:2608.26157 [cs.AI] https://arxiv.org/abs/2608.26157

[52] Anik Pramanik、Murat Kantarcioglu、Vincent Oria 和 Shantanu Sharma。2026。DexterSQL：用于文本转 SQL 生成的深度模式探索与基于规则的纠正。arXiv: 2608 11889 [ ] 2608 11889 2, 。cs.DB 8 2026 https://arxiv.org/abs/。v 九月。[53] Diptikalyan Saha、Avrilia Floratou、Karthik Sankaranarayanan、Umar Farooq Minhas、Ashish R. Mittal 和 Fatma Özcan。2016。ATHENA：一个用于在关系型数据存储上进行自然语言查询的本体驱动系统。*VLDB Endowment* 论文集9，12（2016年），1209–1220。https://doi.org/10.14778/2994509.2994536[54] Irina Saparina 和 Mirella Lapata。2024。AMBROSIA：将有歧义的问题解析为数据库查询的基准测试。载于《神经信息处理系统进展》第*37*卷（*NeurIPS 2024*），数据集与基准测试专题。90600–90628。arXiv:2406.19073[55] Juan Sequeda、Dean Allemang 和 Bryon Jacob。2024。用于了解知识图谱在大语言模型准确率中的作用的基准测试：针对企业级 SQL 数据库问答。载于第七届图数据管理经验与系统联合研讨会（*GRADES*）与网络数据分析（*NDA*）论文集。ACM，1–12。https://doi.org/10.1145/3661304.3661901 arXiv:2311.07509[56] Vladislav Shkapenyuk、Divesh Srivastava、Theodore Johnson 和 Parisa Ghane。2025。面向文本转 SQL 的自动元数据提取。arXiv:2505.19988 [cs.DB] https://arxiv.org/abs/ 2505 19988 。sqlsure。sqlsure：SQL 语义检查器。GitHub 仓库，[57] 2026 https://github.com/sqlsure/sqlsure。[58] Daniel Yitian Su、Sophie Yiran Su、Qiang Sun、Yihao Ding 和 Wei Liu。2026。解耦结构与语义：模式表示如何影响基于大语言模型的 SQL 生成。arXiv:2608.20356 [cs.DB] https://arxiv.org/abs/2608.20356[59] Yoga Sri Varshan Varadharajan、Ajay Yadav、Ritesh Goru、Prateek Chaudhury、Constantine Caramanis、Prateek Jain、Divyateja Pasupuleti 和 Sunil Kumar Pandey。2026。面向嵌套企业模式的自然语言到 SQL 成本感知型智能体架构及一项新基准测试。arXiv:2609.04641 [cs.AI] https://arxiv.org/abs/2609.04641[60] Qin Wang、Youhuan Li、Suixi Lin、Zhuo Tang、Kenli Li、Peng Peng、Quanqing Xu 和 Chuanhui Yang。2026。MCI-SQL：采用元数据完备上下文和中间纠正的文本转 SQL。arXiv:2603.13390 [cs.DB] https://arxiv.org/abs/ 2603 13390 。Niklas Wretblad、Fredrik Riseby、[61] Rahul Biswas、Amin Ahmadi 和 Oskar Holmström。2024。理解文本转 SQL 中噪声的影响：BIRD-Bench 基准测试考察。载于计算语言学协会第*62*届年会论文集（第*2*卷：短论文）。计算语言学协会，356–369。https://doi.org/10.18653/v1/2024.acl-short.34[62] Weipeng P. Yan 和 Per-Åke Larson。1995。即时聚合与惰性聚合。载于第*21*届超大型数据库国际会议论文集（*VLDB*）。瑞士苏黎世，345–357。https://www.vldb.org/conf/1995/P345.PDF[63] Tao Yu、Rui Zhang、Kai Yang、Michihiro Yasunaga、Dongxu Wang、Zifan Li、James Ma、Irene Li、Qingning Yao、Shanelle Roman、Zilin Zhang 和 Dragomir Radev。2018。Spider：用于复杂跨领域语义解析和文本到 SQL 任务的大规模人工标注数据集。载于《*2018*年自然语言处理实证方法会议论文集》（*EMNLP*）。计算语言学协会，3911–3921。https://doi.org/10.18653/v1/D18-1425[64] Tingkai Zhang、Chaoyu Chen、Cong Liao、Jun Wang、Xudong Zhao、Hang Yu、Jian-chao Wang、Jianguo Li 和 Wenhui Shi。2024。SQLfuse：通过全面的大语言模型协同提升文本转 SQL 性能。arXiv:2407.14568 [cs.CL] https://arxiv.org/abs/ 2407 14568 。Ruiqi Zhong、Tao Yu 和 Dan Klein。2020。基[65] 于蒸馏测试套件的文本转 SQL 语义评估。载于《*2020*年自然语言处理实证方法会议论文集》（*EMNLP*）。计算语言学协会， 396 411 10 18653 1 2020 29–。[66] https://doi.org/。/v /。emnlp-main 2026。Tianxin Zhou 和 Ruixi Lin。ModularSQL：针对文本转 SQL 多重性盲区的运行时防护机制。arXiv:2609.29573 [cs.CL] https://arxiv.org/abs/2609.29573

A 证据的第二个示例

图 7 展示了商店问题的证据。图 14 展示了一个供应问题的证据：“哪些产品的订单总数量超过了我们持有的库存数量？”已知表为 products、order_lines 和 inventory；度量 unit_price、quantity 和 on_hand 不纳入统计数据和依赖关系概况。

B 提示（所有实验组共用）

每次运行都会接收以下系统提示和用户消息；换行已重新排版。灰色文本是占位符（图 7 展示了证据块）；蓝色指令仅出现在两个工具组中。

系统提示

你需要使用 `execute_sql` 工具回答有关 SQLite 数据库的问题。你报告的每个数字都必须来自你在本轮中运行的查询。如果查询失败，请根据下方的模式检查列名和表名，并修正错误。不要根据名称或常见情况推断表和列之间的关系：请在数据中核实，未经核实的关系一律视为未知。最后，调用 `submit_answer`，提交能够回答问题的精确 SQL，以及简短的文字答案。提交的 SQL 必须只选取问题要求的内容，不得包含额外列（例如，除非问题要求，否则不要包含计数或 ID）；每个单元格只能有一个值（不得在一列中放置列表或拼接值）。

基数在编写最终 SQL 之前，必须对查询读取的每一张表运行 `基数矩阵` ，并传入查询所依赖的列。查询所依赖的列之间的所有关系，都必须以其结果为依据。若提交内容读取了未运行 `基数矩阵` 的表，则会被拒绝。

数据库模式<所示表的创建表语句>

用户消息

<问题><统计、统计副本、函数依赖组和 Grain-FD 组的证据块+>

C 各模型结果

GrainTraps，每个单元格168次非对照运行，正确 / 粒度错误（表3）。

表3：各模型的 GrainTraps（每个单元格168次非对照运行）：正确 / 粒度错误。粗体：该行粒度错误最少者（并列时均加粗）。

模型	no 证据	统计	统计+Rep	FD	FD（工具）	Grain-FD	Grain-FD（工具）
Claude Sonnet 5.5	165 / 3	**168 / 0**		**168 / 0**	164/ 4	**168 / 0**	**168 / 0**	**168 / 0**
Claude Haiku 5.5	162 / 5	165 /2		165/1	158 /4	165/2	**166 / 0**	**165 / 0**
DeepSeek V4.1 Flash	147 / 18	151 /15		153/11	159 /8	154/11	**166 / 0**	165 / 1
GPT-5.6 Luna	150 / 18	156 /12		152 / 14	152 / 16	148 / 18	161 / 5	**163 / 3**
GPT-5.6 Terra	145 / 21	160 /7		158 / 9	153 / 13	148 / 19	**167 / 0**	162 / 5
GLM-5.3-Flash	130 / 32	142 /18		147 / 12	140 / 13	129 / 24	**157 / 7**	142 / 15
Gemini 3.5 Flash-Lite	88 / 67	101 /52		114 / 35	112 / 44	110 / 42	123 / 39	**126 / 31**
Nemotron 3.5 Super	83 / 70	88 /66		89/63	90 /61	95/60	95 / 61	**106 / 49**

BIRD-FanOut 训练集，229 个失真问题（审核后的标准答案），正确 / 精确扇出错误（答案复现了失真标准答案或商定的陷阱查询）（表4）。

表 4：每个模型在 BIRD-FanOut 训练集（229 个失真的问题）上的结果：正确答案 / 精确扇出错误。粗体表示该行中精确扇出错误最少。

模型	no 证据	统计	统计+Rep	FD	FD（工具）	Grain-FD	Grain-FD（工具）
Claude Sonnet 5.5	151 / 26	162 /18		168/11	159 /20	**172 / 8**	162 / 15	**171 / 8**
Claude Haiku 5.5	**152 / 22**	138 /31		134/32	145 /29	145/29	149 / 29	150 / 28
DeepSeek V4.1 Flash	140 / 27	138 /28		147/25	144 /23	149/21	148 / 20	**150 / 19**
GPT-5.6 Luna	148 / 22	158 /15		168/8	162 /9	160/13	**171 / 5**	164 / 7
GPT-5.6 Terra	142 / 28	153 /22		144 / 25	**153 / 20**	144 / 24	**150 / 20**	**147 / 20**
GLM-5.3-Flash	134 / 37	133 /33		137 / 21	**142 / 19**	129 / 30	145 / 21	130 / 24
Gemini 3.5 Flash-Lite	122 / 45	120 /48		127/37	125 /46	122/43	121 / 42	**133 / 34**
Nemotron 3.5 Super	116 / 29	120 /30		116/32	105 /36	114/30	**114 / 27**	112 / 29



图 14：来自 GPT-5.6 Terra 运行轨迹的第二个 GrainTraps 问题（“哪些产品的订单总数量大于我们持有的其库存量？”，表 inventory）的六种证据实验组（FD（工具）调用来自 Nemotron 3.5 Super），已去除 Markdown 标记，并将多对多箭头绘制为 ↔。行：统计数据、函数依赖和粒度感知函数依赖；列：提示中提供的信息和工具提供的信息。蓝色：某实验组相较于其参考组新增的内容；灰色斜体：模型的工具调用。

BIRD-FanOut 训练集上的扇出错误（精确扇出错误，以及其他被判定为过度计数的答案；§6.3）（表 5）。

表 5：按模型划分，BIRD-FanOut 训练集，扇出错误。粗体：该行中数量最少。

模型	no 证据	统计	统计+Rep	函数依赖	FD（工具）	Grain-FD	Grain-FD（工具）
Claude Sonnet 5.5	31	22	15 24	**10**	20	**10**
Claude Haiku 5.5	**32**	42	42 39	37	33	35
DeepSeek V4.1 Flash	36	39	35 34	28	**26**	31
GPT-5.6 Luna	37	20	12 15	18	**8**	11
GPT-5.6 Terra	43	32	36 30	31	28	**27**
GLM-5.3-Flash	46	47	36 **26**	37	30	39
Gemini 3.5 Flash-Lite	64	65	53 60	60	60	**45**
Nemotron 3.5 Super	48	45	47 52	49	45	**42**

D 补充运行与检查

- **完整模式运行** (§5.2)。§5.2 中的八个模型在展示整个数据库、并将度量纳入证据的情况下，分别运行了无证据、FD 和 FD（工具）三种设置。在 GrainTraps 上，证据使四个较小模型的粒度错误减少了 5–12 点，使四个较大模型减少了 4–5 点，几乎全部发生在 NULL 划分上。在 150 个普通 Mini-Dev 问题上，证据没有造成损害（使用 FD（工具）时的准确率为 +1.0 点 [−2.0, +4 .2]）。

- **设计集**（§5.2）。Grain-FD 规则基于 350 次运行的试点制定：包括 FD 表现不如 Stats 的 GrainTraps问题、NULL 拆分问题，以及 FD 改变 Haiku 或 Luna 结果的 49 个 BIRD 训练集问题（经标准答案审核后剩余 46 个）。试点运行属于报告结果的一部分；其余所有 Grain-FD 运行均在规则确定后进行。

- **标准答案审核**（§4）。在 421 个训练集金标准中，80 个进行了修正（56 个失真，24 个为预期的），37 个被排除：其中 27 个遭到评审员质疑，10 个已损坏或没有扇出信号。12 个接受修正的问题返回了空值、0 或 NULL，并在数据上进行了检查：其中 2 个得到纠正，9 个被排除（答案为 0 或 NULL 无法显示扇出，或问题有歧义），1 个予以保留。

- **对金标准审查的稳健性** (§6.3)。审查在所有运行结束后进行，但 Stats+Rep 的运行除外；这些运行经过了重新评分。审查几乎没有改变扇出结果：在原始标准答案上，Grain-FD 比 Stats 少了 3.9 点扇出错误，审查后少了 3.4 点；Grain-FD（工具）分别少了 4.1 点和 3.9 点。准确率变化更大，因为金标准错误地将正确答案判为错误：在无证据设置下，失真问题的准确率从 49.1% 升至 60.3%，预期问题的准确率从 68.7% 升至 78.8 %。在原始标准答案上，准确率相较 Stats 的提升分别为 +1.9 点（$p = .018$）和 +1.4 点（$p = .20$）。审查后，评审员仍将 6.5% 的运行结果判定为可接受答案，尽管标准答案与之不符；我们将这些结果计为错误。

- **经评审判定的扇出**（§6.3）。在 Gemini 与审计结论不一致时，Opus 大多支持 Gemini：在 Gemini 判定为扇出、但审计未标记的 239个答案中，Opus 确认了 208 个；在审计标记、但 Gemini 未判定为扇出的 20 个答案中，Opus 否定了 16 个。对于与失真标准答案相同的答案，Gemini 将 300 个中的 294 个判定为扇出。将七个实验组所有不同答案合并为一批重新判定后，结果与原始标签的吻合率为 98.7%，且未改变任何结论。

E 完整结果表

这些表格给出了图 8、9、10 和 13 背后的精确数值。比较项按“测试实验组与参考组”列示；粒度和扇出错误以避免的点数表示（正值更好），准确率以提升的点数表示（表6）。

表6：无证据的 GrainTraps：每个模型在168次非对照运行中的正确运行数和粒度错误数，以及执行了检查查询的运行占比（图8）。粗体：各列中的最佳值。

tier	模型	正确	粒度错误	执行了检查查询的运行
低错误	Claude Sonnet 5.5	**165**	**3**	**98%**
低错误	Claude Haiku 5.5	162	5	95%
中等错误	DeepSeek V4.1 Flash	147	18	67%
中等错误	GPT-5.6 Luna	150	18	32%
中等错误	GPT-5.6 Terra	145	21	27%
中等错误	GLM-5.3-Flash	130	32	37%
高错误	Gemini 3.5 Flash-Lite	88	67	21%
高错误	Nemotron 3.5 Super	83	70	11%

表7：GrainTraps，测试实验组避免的粒度错误（点，95%区间），以及粒度错误较少／较多的基础问题（一个问题及其模糊孪生问题计为一个）；适用于所有层级，置换 p（图9，左）。

比较	低错误	中等错误	高错误	all
统计与无证据	+1.8 [+0.0, +4.5], 2/0	+5.5 [+0.9, +11.5], 9/4	+5.7 [−0.6, +12.7], 13/9	+4.6 [+1.3, +8.6], 15/7, 页 = .015
函数依赖与统计	−1.8 [−5.6, +0.0], 0/1	+0.3 [−5.7, +5.8], 8/2	+3.9 [−0.6, +8.0], 13/6	+0.7 [−3.6, +4.0], 15/7, 第 = .78 页
FD（工具）与统计	0.0 [−1.2, +1.5], 1/2	−3.0 [−8.2, +0.8], 5/8	+4.8 [+0.0, +9.8], 13/6	−0.3 [−3.2, +2.0], 12/8, p = .88
Grain-FD 与统计	+0.6 [+0.0, +1.8], 1/0	+6.0 [+1.8, +11.8], 10/1	+5.4 [−0.6, +11.5], 15/6	**+4.5 [+1.3, +8.3], 15/6, 第 = .012 页**
Grain-FD（工具）与统计	+0.6 [+0.0, +1.8], 1/0	+4.2 [−0.1, +10.3], 7/5	+11.3 [+4.5, +18.5], 16/4	**+5.1 [+1.9, +8.9], 15/5, p = .003**
统计+Rep 与统计	+0.3 [+0.0, +0.9], 1/0	+0.9 [−1.7, +3.4], 8/3	+6.0 [+1.2, +10.7], 14/5	+2.0 [+0.6, +3.4], 15/6, p = .019
Grain-FD 与 统计+Rep	+0.3 [+0.0, +0.9], 1/0	+5.1 [+0.6, +11.1], 8/2	−0.6 [−7.2, +5.7], 11/8	+2.5 [−0.6, +6.1], 14/7, 第 = .19 页
Grain-FD（工具）与统计+Rep	+0.3 [+0.0, +0.9], 1/0	+3.3 [−1.2, +9.6], 7/5	+5.4 [−0.6, +12.0], 11/9	+3.1 [+0.1, +6.7], 10/6, p = .089

表 8：GrainTraps，测试实验组的准确率增幅（点，95%区间），随后是答对运行次数增多/减少的基础问题；所有层级均采用置换 p。

比较	低错误	中等错误	高错误	all
统计与无证据	+1.8 [+0.0, +4.5], 2/0	+5.5 [+1.2, +11.0], 12/5	+5.4 [−0.9, +12.6], 13/7	+4.5 [+1.5, +8.3], 16/5, p = .011
函数依赖 vs 统计	−3.3 [−7.6, −0.6], 0/5	−0.7 [−6.7, +4.8], 9/7	+3.9 [−0.6, +8.2], 13/8	−0.2 [−4.4, +3.2], 13/12, 第 = 页94
函数依赖（工具） vs 统计	0.0 [−1.2, +1.5], 1/2	−4.5 [−9.5, −0.4], 5/11	+4.8 [−0.6, +9.9], 16/6	−1.0 [−4.3, +1.5], 13/10, 第 = .58页
Grain-FD 与 统计	+0.3 [−1.2, +1.9], 2/2	+6.2 [+1.8, +12.3], 11/4	+8.6 [+2.3, +15.2], 15/6	**+5.4 [+2.0, +9.2], 16/7, p =.006**
Grain-FD（工具）与统计	0.0 [−1.8, +1.8], 1/1	+3.4 [−1.3, +10.0], 9/8	+12.8 [+5.7, +20.0], 18/3	**+4.9 [+1.4, +8.9], 16/4, 第 = .010 页**
统计+重复与统计	0.0 [−0.9, +0.9], 1/1	+0.1 [−2.9, +3.2], 8/9	+4.2 [−0.6, +8.8], 14/9	+1.1 [−0.9, +3.1], 15/10, p =.34
Grain-FD 与统计+Rep	+0.3 [−1.2, +2.0], 2/2	+6.1 [+1.1, +12.4], 12/4	+4.5 [−2.9, +10.9], 16/7	+4.2 [+0.7, +8.3], 17/7, p =.045
Grain-FD（工具）与统计+Rep	0.0 [−1.5, +1.2], 2/1	+3.3 [−1.5, +9.8], 10/9	+8.6 [+1.8, +15.5], 14/8	+3.8 [+0.3, +8.0], 13/7, p =.067

表 9：BIRD-FanOut 训练集（229 个失真问题，每个实验组 1,832 次运行），测试实验组避免的扇出错误（点；正值更好）、扇出运行次数较少／较多的问题；所有层级均列出 95%区间和置换 p；层级仅作描述（图 9，右侧）。

比较	低错误	中等错误	高错误	all
统计与无证据	−0.2, 8/12	+2.6, 38/25	+0.4, 23/20	+1.4 [−0.1, +2.9], 49/39, p = .087
函数依赖 vs 统计	+0.2, 11/10	+3.6, 39/13	−0.4, 22/24	+1.7 [+0.3, +3.2], 48/30, p = .028
FD（工具）与统计	+3.7, 22/6	+2.6, 45/23	+0.2, 25/24	+2.3 [+0.8, +3.7], 58/33, p 值为 = .004
Grain-FD 与统计	+2.4, 19/9	+5.0, 43/12	+1.1, 26/21	**+3.4 [+1.9, +4.9], 59/26, p < .001**
Grain-FD（工具）与统计	+4.1, 25/8	+3.3, 40/19	+5.0, 32/11	**+3.9 [+2.2, +5.6], 67/29, p < .001**
统计+Rep 与统计	+1.5, 13/7	+2.1, 36/21	+2.2, 26/20	+2.0 [+0.5, +3.4], 49/35, p = .013
Grain-FD 与统计+Rep	+0.9, 12/9	+2.9, 34/14	−1.1, 25/28	+1.4 [+0.3, +2.6], 54/28, 第 = 页 .024
Grain-FD（工具）与 Stats+Rep	+2.6, 17/6	+1.2, 32/26	+2.8, 35/20	+2.0 [+0.6, +3.4], 57/29, p = .007

表10：BIRD-FanOut 训练集中的扇出错误；相较于统计数据，使用和不使用大语言模型评审器时避免的错误，以及粒度错误（点）；扇出运行较少/较多的问题、置换 p；全部229个失真问题和183个留出问题（图10）。

扇出指标	Grain-FD 与统计数据对比，229	Grain-FD（工具）与统计数据对比，229	Grain-FD 与统计数据对比，183	Grain-FD（工具）与统计数据对比，183
仅精确	+2.5，44/18，p < .001	+3.1，46/19，p < .001	+1.8，31/14，p = .008	+2.4，32/15，p = .002
精确 +审计标记	+3.1，57/25，p < .001	+3.3, 59/27, p < .001	+2.2，42/21，p = .004	+2.5，43/23，p = .002
精确 +判定（主要）	+3.4，59/26，p < .001	+3.9，67/29，p < .001	+2.7，44/21，p < .001	+2.9，49/27，p = .001
粒度错误（精确 +经评审判定的扇出或粒度错误粒度）	+3.1，60/31，p < .001	+3.7，66/34，p < .001	+2.4，45/25，p = .007	+2.7，48/30，p = .004

表11：BIRD-FanOut 训练集，183 个留出问题中各实验组的扇出错误数（每组 1,464 次运行；图10）。粗体：数量最少。

问题	无证据	统计	FD	函数依赖（工具）	Grain-FD	Grain-FD（工具）
留出 183 个	236	221	204	189	181	**179**

表12：“其他”陷阱答案的原因（Opus 判断器；完整模式运行；图13）。

原因（Opus 判断器）	不同答案
粒度错误（按与问题所要求不同的粒度排名或分组）	166
误解问题	53
SQL错误（其中 20 个涉及 NULL处理）	44
与朴素SQL不匹配的过度计数	43
低估计数：被当作重复项移除的真实行	20
标准答案不接受的合理解读	1