

产品链接：

[https://www.amazon.com/gp/product/B0BGHH2L23?
th=1](https://www.amazon.com/gp/product/B0BGHH2L23?th=1)

任务：请站在美国用户的视角，当你需要购买该产品时，你会关注哪些信息点，原因是什么（1~2句话说明即可），要点式展示（示例：关注点：原因）

已思考 9s >

好的！站在美国用户视角，购买该产品（iSPECLE 可伸缩水槽置物沥水架）时常见关注点如下（“关注点：原因”）：

- 尺寸与适配：需要确认是否适合自家水槽（支持 12.4"-17.1" 可伸缩，宽 12.8"，高 5.7"），避免退换麻烦。[亚马逊](#)
- 可伸缩/安装位置：是否能放“水槽内/跨水槽/台面上”三用，提升小厨房与 RV 的空间利用率。[亚马逊](#)
- 防锈与材质：金属+防锈涂层是否耐用、好清洁，减少长期使用后的生锈和异味。[亚马逊](#)
- 稳定性与承重：是否有止滑脚垫、L 型支撑设计与标注承重（页面写 39 磅），避免倾倒伤盘碟。[亚马逊](#)
- 容量与分区：能放几只盘/碗/杯，是否含可拆刀叉篮，满足 1-3 人日常。[亚马逊](#)
- 安装与固定：是否需要工具/螺丝，调好后会不会滑动，提升使用安全感。[亚马逊](#)
- 尺寸提示与限制：页面特别提示“不适合 12x14 水槽”，需提前量尺避免踩雷。[亚马逊](#)
- 外观与颜色：是否有与厨房风格匹配的配色（黑色等），家居一致性更好。[亚马逊](#)
- 价格与优惠：是否有优惠券/低价卖家选项，提升性价比（页面有 10% coupon、其他卖家更低价）。[亚马逊](#)
- 运输与退货：Prime 配送速度、是否“免费退货/延长假期退货”降低购买风险（2025 假期可退至 2026-01-31）。[亚马逊](#)
- 销量与口碑：近期销量“过去一个月 600+”这类社证明确降低决策焦虑。[亚马逊](#)
- 品牌与店铺：是否“由 iSPECLE 销售/亚马逊发货”，官方店铺与 FBA 让售后更有保障。[亚马逊](#)
- 清洁与维护：是否易冲洗、是否可拆分，减少污垢堆积与霉味。
- 台面保护：止滑脚垫是否会刮花水槽或台面，影响长期使用体验。[亚马逊](#)
- 重量与移动性：产品重量 1.9 磅，是否利于拿起清洁或在不同位置切换。[亚马逊](#)
- 包装方式：是否“原厂包装直寄/可选亚马逊外箱”，对送礼或隐私有影响。[亚马逊](#)

- 对比替代：是否有同尺寸/不锈钢材质/双层设计的可选品，方便快速横向比较。
- 问答与图片：是否有实拍、尺寸示意图与 FAQ，帮助快速确认兼容性。 亚马逊

(以上要点依据该商品页的参数、功能描述、优惠与政策信息整理。)



假设你已经收到该产品，但是你所关注的点都不符合预期，注意是上述所有的点噢，你要退货，每一个关注点的退货留言会如何写呢？每个关注点的留言用2~3句话表达不符合预期即可，需要有说服力的

好的，按“关注点：退货留言（2-3 句）”给你逐条写好，均以美国买家的语气、可说服为主：

- 尺寸与适配：量了我家水槽（内宽 16.5"），实物仍然放不稳，两侧支撑会滑进槽里。页面标注的伸缩范围与实际可用范围不一致，导致无法正常使用。
- 可伸缩/安装位置：我需要“跨槽/槽内/台面”三种摆放，但它在槽内角度会倾斜，跨槽时一端总是下坠。这和页面展示的多场景适配不符。
- 防锈与材质：开箱 3 天内接触清水就出现锈斑与水渍难去的问题。若是“防锈涂层”，耐久度明显达不到日常厨房使用标准。
- 稳定性与承重：宣称高承重，但放入 2 个陶瓷盘和一口小锅就轻微下弯并晃动。防滑脚垫抓力不够，转角处会移动，存在安全隐患。
- 容量与分区：刀叉篮偏小且开口太窄，普通餐刀插入会顶到边。整体容量比宣传图少，无法满足两口之家一餐的晾放量。
- 安装与固定：无需工具是优点，但调节后固定性差，轻碰就回弹。说明书对如何稳固使用的指导不足，无法达到即装即用的体验。
- 尺寸提示与限制：页面提示“不适合 12x14 水槽”，我家尺寸更大却仍不适配。说明中的限制条件不够清晰，实际兼容范围被高估。
- 外观与颜色：选择黑色是为了与厨房搭配，但表面涂层不均、边角喷涂发灰。与商品图存在明显色差，影响台面观感。
- 价格与优惠：下单时显示有 coupon，结算页未自动应用；同页面“其他卖家更低价”下单后才看到。性价比与预期不符，价格透明度不足。
- 运输与退货：我选择了 Prime 以获得更快配送，但比预计时间晚了两天，且外箱有明显压痕。以防损为由我必须尽快退货。

- 销量与口碑：基于“本月热卖”等社证明我才下单，但实际体验不达标。该社证与实际品质不一致，影响我的信任。
- 品牌与店铺：页面显示由品牌/亚马逊履约更有保障，但客服响应慢、换货流程不清晰。售后与页面承诺存在落差。
- 清洁与维护：格栅间距与死角多，食物残渣容易卡住，冲洗后仍留水印。日常维护成本超出预期。
- 台面保护：脚垫材质偏硬，第一次放到石英石台面就留下细微划痕。宣称的防滑/防刮效果没有达到。
- 重量与移动性：产品比标注的更沉，频繁在“槽内一台面”间移动很费力。对想要灵活挪放的用户不友好。
- 包装方式：收到的是原厂裸盒，角落破损，内部没有足够防护。金属表面出现磕痕，像是运输/包装保护不到位。
- 对比替代：我选它是因为“可伸缩+单层省空间”的综合方案，但实际稳定性与容量都不如同价位不锈钢款。综合比较后，这款不具备优势。
- 问答与图片：商品问答与买家实拍显示“适配大多数单槽”，但与我的实际情况差距很大。页面图片缺少关键尺寸示意，容易误导判断。

以上每条都可单独作为退货理由配合订单信息提交，既具体又可验证。需要我把这些合并成一段更标准的 RMA 说明模板吗？



我还需要一版英文版的

Got it—here's the English version, in the same "Focus: return note (2–3 sentences)" style:

- Size & fit: I measured my sink (16.5" inner width), but the rack still doesn't sit securely—both sides slide inward. The usable range doesn't match what's advertised, so I can't use it as intended.
- Expandability/placement: I needed to use it in-sink, over-sink, and on the counter, but it tilts in-sink and sags when spanning the basin. That's not what the listing photos suggest.
- Anti-rust & material: Within three days of normal use, rust spots and stubborn water marks appeared. If there's an anti-rust coating, it isn't holding up to basic

kitchen use.

- Stability & load: It's marketed as high-capacity, yet two ceramic plates and a small pot make it bow and wobble. The anti-slip feet don't grip well, creating a safety concern.
- Capacity & sections: The utensil caddy is too narrow—standard dinner knives hit the rim. Overall capacity is less than shown and doesn't handle a typical meal for two.
- Setup & fastening: Tool-free is nice, but once adjusted it doesn't stay put—light bumps make it shift back. The instructions don't explain how to keep it stable for daily use.
- Sizing warnings/limits: The page warns against 12" × 14" sinks; mine is larger and it still doesn't fit properly. The real compatibility range seems overstated.
- Look & color: I chose black to match my kitchen, but the finish is uneven and edges look grayish. The color doesn't match the product photos and hurts the overall look.
- Price & deals: A coupon showed on the page but didn't apply at checkout, and I noticed a lower "other seller" price only after ordering. The value and pricing transparency weren't as expected.
- Shipping & returns: I paid for Prime speed, but it arrived two days late with a crushed corner on the box. Given the risk of damage, I'm returning it.
- Sales proof & reviews: I trusted the "bestseller/600+ this month" type signals, but the quality doesn't reflect that. The social proof feels misleading given the actual performance.
- Brand/store support: It's listed as brand-sold/FBA for better support, yet customer service was slow and the exchange process unclear. Post-purchase help didn't match the promise.
- Cleaning & maintenance: The grate spacing traps food bits and there are many hard-to-rinse corners. Even after washing, water spots remain—maintenance is higher than expected.
- Counter protection: The feet are quite firm and left faint scratches on my quartz countertop on first use. The anti-slip/anti-scratch claims didn't hold up.
- Weight & mobility: It feels heavier than listed and is awkward to move between sink and counter. Not great if you need to reposition it frequently.
- Packaging: Arrived in a thin retail box with a torn corner and minimal padding. The metal had scuffs out of the box—packaging protection was inadequate.

- Alternatives vs. value: I chose this for the “extendable + compact” combo, but it's less stable and holds less than similarly priced stainless models. It doesn't offer an advantage overall.
- Q&A & photos: The Q&A and user photos imply it fits most single sinks, but that wasn't my experience. The listing lacks critical dimension diagrams, which led to a bad fit decision.

If you want, I can consolidate these into a single, professional RMA paragraph or a template you can paste into Amazon's return form.



角色切换，现在你是一名资深的亚马逊美国站点的产品总监，同时也是运营总监，请你根据上述的关注点，退货留言，搭建退货留言的标签库，标签库需要遵循MECE原则，要点式展示（示例：标签（中文）：定义（注意相似标签之间的边界划分））

已思考 30s >

好的，我以“产品总监/运营总监”的视角，基于前述关注点与退货留言，构建一套遵循 MECE 原则的【退货留言标签库】。每条均给出清晰定义，并标注与近似标签的边界：

- 尺寸/兼容性不符：实物与水槽尺寸或开口条件不匹配，无法稳固放置或无法装下。
(仅涉及“能否合适放置/装得下”，不含晃动承重问题)
- 场景适配不足（摆放模式）：无法满足“槽内/跨槽/台面”等多场景的使用方式。（若能放下但晃动，请选“稳定性/承重不足”；若因尺寸不合，请选“尺寸/兼容性不符”）
- 规格参数与宣传不一致：伸缩范围、承重、重量、尺寸限制等与页面标称不符。（若是颜色/表面观感不符，选“外观/颜色与实物不符”）
- 稳定性/承重不足：使用中出现下弯、晃动、打滑，无法安全承载日常餐具锅具。（台面被刮请选“台面/器具保护不足”）
- 容量不足：实际可放置数量/分区（如刀叉篮）低于宣传或无法满足家庭一餐需求。
(结构稳固性不足不在此类)
- 材质/防锈失效：短期使用即生锈、涂层脱落或水渍难清，耐久性不符预期。（色差、喷涂不均属“外观/颜色与实物不符”）

- 外观/颜色与实物不符：色差明显、涂层不均、边角发灰等影响观感。（与耐用性相关的问题归“材质/防锈失效”）
- 台面/器具保护不足：脚垫或接触面造成划痕，防滑/防刮效果不达标。（整体晃动安全性不足，请选“稳定性/承重不足”）
- 安装/调节困难：调节后难以固定、易回弹，说明指引不足影响上手。（若调好后仍晃动，属“稳定性/承重不足”；若尺寸本身不合，属“尺寸/兼容性不符”）
- 清洁/维护成本高：缝隙易积污、死角多、水印难去，日常保养负担大。（锈蚀归“材质/防锈失效”）
- 重量/移动不便：实际偏重或抓握不便，频繁在不同位置移动费力。（放不下或放不稳定不在此类）
- 定价/促销不透明：优惠券未生效、下单后发现同页更低价等导致性价比不及预期。（若仅为整体价值感偏低，请选“价值/对比劣势”）
- 价值/对比劣势：与同价替代品相比，综合稳定性/容量/材质等无优势，性价比不高。（若能指向具体缺陷，请优先在相应功能标签打标，避免重复）
- 物流时效异常：Prime 或承诺时效未达，延迟到货影响使用。（若包装损坏但未延迟，到“包装防护不足”）
- 包装防护不足：外箱/内衬防护不充分，导致磕碰痕迹或影响开箱体验。（若商品本体已损坏影响使用，可与“损坏到货”二择一，主诉求优先）
- 损坏到货（DOA/受损）：开箱即有明显损伤或功能性问题，无法正常使用。（若损伤源于外包装破损但商品可用，优先“包装防护不足”）
- 商品页面信息缺失/误导：缺少关键尺寸示意、要点说明不清，或 Q&A/买家秀造成错误判断。（若为参数数值不符，选“规格参数与宣传不一致”）
- 社会证明与体验不符：销量/榜单/标签的“热卖”形象与实际体验差距大，影响信任。（不讨论评论真假，仅聚焦“社证与体验脱节”）
- 品牌/店铺服务不达标：客服响应慢、换退流程不清晰、承诺与实际售后不一致。（物流延迟归“物流时效异常”，退货规则问题见下一条）
- 退换/政策体验不佳：退换流程复杂、周期/规则与承诺不符，影响退货体验。（与客服响应速度区分，避免与上一条重复）

这套标签库做到“互斥且穷尽”：每条退货留言可在不重叠的前提下映射到唯一主标签；如需细化运营分析，建议在系统中为每个主标签配置二级原因码（如“容量不足/刀叉篮偏小”“容量不足/主架有效面积不足”），以便后续做定量诊断与改版优先级排程。



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

Easy to assemble. Mostly fits the sink, but there is still space left open on the sides. I understand that it won't be perfect since it depends on the sink. I found the utensil holder too large to fit well. It should be shorter so it stays in place.

已思考 21s >

- **留言翻译：**

组装很容易。基本能放进水槽，但两侧仍有空隙。我理解这取决于水槽，不可能完美贴合。餐具篮太高太大，放不稳；如果更矮一些就能固定住。

- **舆情分析：**

中性偏负。核心问题是“尺寸/兼容性不佳”与“稳固性不足（配件高度导致不稳）”；虽认可“易组装”，但适配与稳定问题更影响使用体验，存在退货风险。

- **退货标签：原文依据**

- **尺寸/兼容性不符**：证据——“Mostly fits the sink, but there is still space left open on the sides.”（基本合适但两侧留空，未能与水槽良好贴合）
- **稳定性/承重不足（配件稳固性）**：证据——“I found the utensil holder too large to fit well... It should be shorter so it stays in place.”（餐具篮过大/过高导致放不稳，需要更矮才能固定）
- **安装/调节困难（不适用/正向反馈）**：证据——“Easy to assemble.”（易于组装，非负向标签，仅用于说明不涉及安装问题）



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在

多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

I find that this item is a bit flimsy. I am constantly having to re tighten the screws and have to move the rubber pieces back in place. Overall it helps keep my sink a bit more organized but I will be searching for a replacement.

已思考 22s >

- **留言翻译：**

我觉得这件商品有点单薄。螺丝我得不停地重新拧紧，橡胶件也总要推回原位。整体确实让水槽更整洁一些，但我会去找替代品。

- **舆情分析：**

负面为主、夹带一条正面点（可稍微提升整洁度）。核心痛点是“稳固性差/固定不住”，导致频繁维护；用户已明确有换购意图，存在退货/转化流失风险。

- **退货标签：原文依据**

- **稳定性/承重不足：**依据——“**a bit flimsy**”（结构感受单薄）+“**have to move the rubber pieces back in place**”（防滑/缓冲件易位，整体不稳）。
- **安装/调节困难（固定性差）：**依据——“**constantly having to re-tighten the screws**”（调好后仍松动、需反复紧固），说明调节后的保持性不足。
- **价值/对比劣势：**依据——“**Overall it helps... but I will be searching for a replacement.**”（有一定作用但仍决定找替代品，体现综合价值不具优势）。



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

Cups fall over when you use the cup holder part on the side. There's no bar that keeps regular sized tall cups from tipping over.

- **留言翻译：**

用侧边的杯架时杯子会倒。没有能固定普通高杯的横杆，杯子容易倾倒。

- **舆情分析：**

明确负面，直指功能性与安全性问题（易倾倒）。属于结构/稳固性缺陷，影响实际可用容量，用户有强退换动机。

- **退货标签：原文依据**

- **稳定性/承重不足（配件稳固性）**：依据——“**Cups fall over when you use the cup holder part on the side.**”（使用杯架时杯子会倒）；“**There's no bar that keeps... cups from tipping over.**”（缺少防倾倒支撑）。
- **商品页面信息缺失/误导**（若页面未标明适配杯高/杯径或需要横杆支撑）：依据——“**regular sized tall cups**”仍会倾倒，属常见杯型的基本适配预期未被满足且页面未明确限制。



上述内容归属于产品A：B0BGHGXYJX

下面开始讨论产品B：B0D6YNTRTZ

产品链接：

[https://www.amazon.com/gp/product/B0CSMK924R?
th=1](https://www.amazon.com/gp/product/B0CSMK924R?th=1)

角色切换，请站在美国用户的角度思考，当你需要购买该产品时，你会关注哪些信息点，原因是什么（1~2句话说明即可），要点式展示（示例：关注点：原因）

已思考 35s >

先说清楚：我以下依据你提供的商品链接（ASIN：B0CSMK924R）页面信息整理；若与B0D6YNTRTZ为不同变体/商品，请以实际下单页为准。

- 尺寸/适配：实测水槽内长需 10.8"-17.9"，内宽 > 12.8"，并预留水龙头/开孔 ≥ 0.6"；不量尺很容易不合适。
亚马逊
- 放置场景：是否能在“槽内/跨槽/台面”三种方式使用，适合小厨房或 RV 的灵活场景。
亚马逊
- 真正可扩容：整机扩展而非仅拉手柄，官方称可在 6 只盘碗之外再稳放至少 3 只杯子，关注实际容量是否匹配家庭人数。
亚马逊

- 材质与防锈：不锈钢+扁铁主框架标称“100% 防锈”，长期接水环境更看重耐腐与易清洁。亚马逊
- 稳定性设计：90° L 形悬挂、双接触支撑，挂臂止滑珠与防滑脚垫，关注防倾倒与防刮台面的实际效果。亚马逊
- 安装与固定：需要简易螺丝安装并通过锁紧防位移；看是否“装好就稳”，减少日常调整。亚马逊
- 承重与自重：标注承重 39 lb、净重约 2.27 lb；承重关系安全感，自重影响挪放与清洁便利。亚马逊
- 刀叉篮与分区：是否带可拆卸刀叉篮，便于分类晾放并提升单位面积的利用率。
- 变体/尺寸选择：同款有 Small/Medium/Large、带杯架/硅胶垫等版本，先比对自家水槽与需求再选型。
- 退货与价格：页面显示“可免费退货”，且可能“其他卖家更低价”；这关系到购买风险与价格透明度。
- 口碑社证：近月“200+ 购买”可作参考，但不等于质量保证，仍需结合真实评价点。
- 台面保护：防滑橡胶脚垫声称可防刮，关注与自家台面材质（石英石、不锈钢等）的适配性。

(如需，我可以把以上要点转成“下单前自检清单”或“选型对比表”。)



假设你已经收到该产品，但是你所关注的点都不符合预期，注意是上述所有的点噢，你要退货，每一个关注点的退货留言会如何写呢？每个关注点的留言用2~3句话表达不符合预期即可，需要有说服力的

好的！按你之前为产品B (B0D6YNTRTZ/B0CSMK924R链接页) 梳理的关注点，我分别写出可直接用于亚马逊退货申请的“2-3句”英文风格（中文表达）留言，逐条覆盖且有说服力：

- **尺寸/适配**：我按商品页的量尺方法测量过水槽，尺寸在标称范围内，但实物仍然两侧留缝且放不平。这导致无法稳固使用，和页面对适配性的描述不一致。
- **放置场景（槽内/跨槽/台面）**：实际在“槽内”会倾斜，“跨槽”时一端下坠，“台面”使用又容易积水回流。与页面展示的多场景适配不相符，无法满足我家日常使用。

- **真正可扩容（有效容量）**：宣传说能在常规放置之外再稳放额外杯具，但实际可用面积有限，杯盘相互干扰。对于两人一餐都不够用，和扩容预期不符。
- **材质与防锈**：正常清洗与沥水后不久就出现斑点与水痕难以清理，边角涂层也容易磨损。若标称防锈与易清洁，耐久度明显没有达标。
- **稳定性设计（防滑/防倾倒）**：防滑脚垫与挂臂止滑并不能有效固定，轻碰就位移。盘杯几次出现倾倒风险，使用不安心。
- **安装与固定**：虽不复杂，但调好后并不牢靠，螺丝需要反复紧固才能暂时稳定。说明书对“如何保持不位移”的指导不足，达不到“装好就用”的体验。
- **承重与自重**：放入日常餐具后主架有下弯与轻微晃动；同时整机偏重，来回在“槽内—台面”之间挪动费力。承重与便携性都低于预期。
- **刀叉篮与分区**：刀叉篮尺寸与开口比例不合理，常用餐刀会顶到边、易晃动。分区的有效利用率不高，反而占了宝贵空间。
- **变体/尺寸选择**：我按页面提示选择了对应变体与尺寸，但到手仍不匹配实际需求。变体间的差异点与适配边界不够清晰，容易误选。
- **退货与价格（透明度）**：结算时优惠未按页面提示自动应用，下单后又看到同页面其他卖家更低价。价格与退货信息不够透明，影响购买信心。
- **口碑社证**：我看重“近期购买量/好评标签”才下单，但实际体验与这些社证不一致。该差距削弱了我对产品质量与宣传的信任。
- **台面保护**：第一次放到石英石台面就出现轻微印痕/划痕，防滑与防刮效果没有达到承诺。为避免进一步损伤台面，我选择退货。

需要的话，我可以把这些条目再整理成「一段式RMA模板」或「多选原因清单」，方便你在退货页面快速勾选与粘贴。



角色切换，现在你是一名资深的亚马逊美国站点的产品总监，同时也是运营总监，请你根据上述的关注点，退货留言，搭建退货留言的标签库，标签库需要遵循MECE原则，要点式展示（示例：标签（中文）：定义（注意相似标签之间的边界划分））

已思考 22s >

好的！基于产品B（对应你前述的关注点与退货留言），我以“产品/运营总监”视角给出一套遵循 MECE 的【退货留言标签库】——每个标签均给出清晰定义，并标注与相近标签的边界，便于一键打标与数据归因：

- **尺寸/兼容性不符**：实物在标称尺寸范围内仍无法良好贴合或稳放于水槽。边界：若能放下但在特定摆放方式（槽内/跨槽/台面）出现问题，归“场景适配不足”；若为数值与页面不一致，归“规格参数与宣传不一致”。
- **场景适配不足（槽内/跨槽/台面）**：在任一或多种摆放场景下出现倾斜、下坠、积水回流等，导致无法按场景稳定使用。边界：若根因是尺寸放不下，归“尺寸/兼容性不符”；若能放下但易晃/易倒，优先“稳定性/防倾倒不足”。
- **规格参数与宣传不一致**：伸缩范围、承重、重量、可容纳数量/额外杯位等硬指标与页面标称不符。边界：若信息未给出或表达不清，归“商品页面信息缺失/误导”；若表现为锈蚀/涂层问题，归“材质/防锈失效”。
- **稳定性/防倾倒不足**：使用中出现下弯、晃动、杯盘易倒或止滑失效等，影响安全性与体验。边界：因承重超限产生的结构形变可在本标签记录；若对台面造成划痕，另标“台面保护不足”。
- **安装/调节/固定性差**：虽可装配，但调好后易位移、需频繁紧固或锁不住位置。边界：若即便锁紧仍晃/倒，另加“稳定性/防倾倒不足”；若根因是尺寸不合，优先“尺寸/兼容性不符”。
- **容量/有效面积不足（含扩容效果不达标）**：实际可用面积与可放置数量达不到宣传或家庭基本需求（如杯盘互相干扰）。边界：因分区/配件尺寸设计不当导致单一部件不可用，另标“分区/配件设计不合理”。
- **分区/配件设计不合理（刀叉篮/杯架等）**：刀叉篮过高/过窄、杯架缺少防倾杆等，导致配件本身难以稳固或降低整体有效容量。边界：如果是整体架体晃/倒，优先“稳定性/防倾倒不足”。
- **材质/防锈失效**：短期使用即出现锈斑、涂层磨损、腐蚀等耐久性问题。边界：若主要抱怨水渍、污渍难清而非锈蚀，归“清洁/维护成本高”。
- **清洁/维护成本高**：易留水印、易积污、死角多、难以冲洗干净，导致维护负担大。边界：若为锈蚀或涂层问题，归“材质/防锈失效”。
- **自重过大/移动不便**：整机过重或抓握不便，在“槽内—台面”等场景切换费力。边界：若移动后出现晃动/倾倒，再加“稳定性/防倾倒不足”。
- **变体选择指引不清**：不同尺寸/版本差异点、适配边界未解释清楚，导致误选。边界：若是标称数值错误，归“规格参数与宣传不一致”；若是缺少必要图示/说明，亦可叠加“商品页面信息缺失/误导”。

- **台面保护不足（防滑/防刮）**：脚垫或接触面在正常使用下留下印痕/划痕，或防滑力不足。边界：若核心问题是整体易倒，优先“稳定性/防倾倒不足”。
- **价格/促销不透明**：优惠券未生效、下单后才发现同页更低价等，造成价格感不佳。边界：若抱怨主要在退换流程/承诺，与价格无关，归“退换信息/政策透明度不足”。
- **退换信息/政策透明度不足**：“可免费退货”等承诺与实际流程/规则不一致或说明不清。边界：物流时效/包装问题不在本标签（如涉及应独立建“物流/包装”类，则在跨品类场景再启用）。
- **社会证明与体验不符**：热卖/好评/榜单等社证与到手体验差距大，削弱信任。边界：不讨论评论真假，仅记录“社证期待值”与“实际表现”的落差。
- **商品页面信息缺失/误导**：缺失关键尺寸图、场景限制、安装注意事项或扩容条件等，导致用户作出错误决策。边界：若页面有明确数值但与实物不符，归“规格参数与宣传不一致”。

使用建议：每条用户留言仅打一个**主标签**，必要时叠加 1–2 个**从标签**（如“稳定性/防倾倒不足” + “分区/配件设计不合理”），避免同层级重复。后续可为各主标签配置二级原因码（例：“配件-刀叉篮过高”“配件-杯架缺防倾杆”），以支持精细化改版与优先级排序。



请你基于产品A和产品B的退货标签库，整合成终版标签库，标签库需要遵循MECE原则，要点式展示（示例：标签（中文）：定义（注意相似标签之间的边界划分）），注意不同产品之间的标签，基于定义来说可能会重合，此时你需要去重处理，但也有可能存在互相补充的情况，此时需要新增，但注意边界划分即可

下面是基于产品A与产品B整合去重后的【终版退货留言标签库】（MECE），每条含：标签 | 定义 | 与相近标签的边界。

一、适配与场景

- **尺寸/兼容性不符**：与标称适配范围内仍无法稳固放置或装入。边界：若能放下但在特定摆放模式出问题，归“场景适配不足”；若页面数值本身不对，归“规格参数与宣传不一致”。

- **场景适配不足（槽内/跨槽/台面/RV 等）**：在某/多场景下倾斜、下坠、积水回流等。
边界：根因是尺寸放不下→“尺寸/兼容性不符”；能放下却晃/易倒→“稳定性/防倾倒不足”。
- **变体/选型指引不清**：尺寸/版本差异点和适配边界说明不清导致误选。**边界**：页面有明确数值但写错→“规格参数与宣传不一致”；缺少关键图示→可叠加“页面信息缺失/误导”。

二、结构与承载

- **稳定性/防倾倒不足**：下弯、晃动、杯盘易倒、防滑失效等影响安全。**边界**：对台面留下划痕→“台面/器具保护不足”；仅某配件设计导致不稳→“分区/配件设计不合理”。
- **容量/有效面积不足（含扩容效果不达标）**：实用容量与宣传或家庭基本需求不符。**边界**：因配件尺寸/结构造成单一部位不可用→“分区/配件设计不合理”。
- **分区/配件设计不合理（刀叉篮/杯架等）**：配件过高/过窄/缺止倾结构，导致难以稳固或压缩有效容量。**边界**：整体架体晃→“稳定性/防倾倒不足”。
- **自重过大/移动不便**：整机偏重或抓握不便，影响在多位置间切换。**边界**：移动后出现晃动→并标“稳定性/防倾倒不足”。

三、材质与外观

- **材质/防锈失效**：短期即锈蚀、涂层脱落、腐蚀等耐久问题。**边界**：仅水渍/污渍难清→“清洁/维护成本高”；色差/喷涂不均→“外观/颜色与实物不符”。
- **外观/颜色与实物不符**：色差明显、涂层不均、边角发灰等影响观感。**边界**：涂层耐久性问题→“材质/防锈失效”。
- **台面/器具保护不足（防滑/防刮）**：脚垫或接触面留下印痕/划痕或防滑力不足。**边界**：若核心是整体易倒→“稳定性/防倾倒不足”。
- **清洁/维护成本高**：易积污、死角多、水印难去，日常保养负担大。**边界**：锈蚀/涂层脱落→“材质/防锈失效”。

四、安装与使用体验

- **安装/调节/固定性差**：装配虽易，但调好后易位移、需频繁紧固或锁不住。**边界**：锁紧后仍晃→“稳定性/防倾倒不足”；根因尺寸不合→“尺寸/兼容性不符”。

五、信息与预期管理

- **规格参数与宣传不一致**：伸缩范围、承重、重量、可容纳数量/额外杯位等硬指标与页面不符。**边界**：页面缺少关键信息→“页面信息缺失/误导”。
- **页面信息缺失/误导**：缺关键尺寸图、场景限制、安装注意/扩容前提，或Q&A引导偏差。**边界**：若已给出明确数值但错误→“规格参数与宣传不一致”。

- **社会证明与体验不符：**热卖/榜单/高分等社证与到手体验落差大。**边界：**不评判评价真假，仅记录“社证预期”与“实际表现”差距。

六、价格与政策

- **价格/促销不透明：**优惠券未应用、同页更低价下单后才可见等导致价格感不佳。**边界：**整体“值不值”感受→“价值/对比劣势”。
- **价值/对比劣势：**与同价替代品相比综合无优势，用户表达更换/弃购倾向。**边界：**若能指向具体缺陷，优先打具体功能性标签，避免重复。
- **退换信息/政策体验不佳：**退换流程复杂、周期/规则与承诺不符或指引不清。**边界：**客服响应慢/处理差→“品牌/店铺服务不达标”。

七、履约与服务

- **物流时效异常：**承诺时效未达、延迟到货。**边界：**包装完好但晚到→仅此标签；包装问题另归“包装防护不足/损坏到货”。
- **包装防护不足：**外箱/内衬防护弱，影响开箱或造成轻微外观问题。**边界：**商品本体功能受损→“损坏到货”。
- **损坏到货 (DOA/受损) :** 到手即明显损伤或功能性问题，无法正常使用。**边界：**若仅包装受损但商品可用→“包装防护不足”。
- **品牌/店铺服务不达标：**客服响应慢、处理不清或与承诺不一致。**边界：**与退换规则本身相关→“退换信息/政策体验不佳”。

使用建议（避免重叠）：

- 为每条留言选1个**主标签**（最核心原因），必要时叠加1-2个**从标签**（直接相关的次要原因）。
- 若留言含“具体功能缺陷 + 综合不值”两类表述，先打功能性标签，再酌情追加“价值/对比劣势”。
- 可为关键主标签配置二级原因码（如“配件-刀叉篮过高”“杯架-缺防倾杆”“脚垫-材质过硬”）以支持精细化复盘与改版优先级。



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会

存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

I recently purchased this dish rack, and while it serves its purpose, I was a bit disappointed. The material feels a bit cheap for the price, which makes me question its durability over time. Additionally, the rack didn't fit my sink properly because my faucet is a bit too wide at the bottom. It's also a bit flimsy, and I don't feel confident it will hold up for long with heavy dishes. Overall, it's functional but not quite what I expected given the cost

已思考 17s >

- **留言翻译：**

我最近买了这个沥水架，虽然能用，但有点失望。以这个价格来说，材质手感有点廉价，让我担心长期耐用性。另外因为我家水龙头底部偏宽，架子并没有很好地适配水槽。它也有点单薄，放重一点的餐具我不太有信心能撑得住。总体能用，但考虑到价格，不如我的预期。

- **舆情分析：**

中度负面（功能可用但不满意）。主要集中在“兼容性不合”“结构稳固性弱”“价格-价值不匹配”三项；存在换购/退货倾向。

- **退货标签：原文依据**

- **尺寸/兼容性不符：**依据——“the rack **didn't fit my sink properly** because my **faucet is a bit too wide at the bottom.**”（因水龙头底座过宽导致不匹配）
- **稳定性/防倾倒不足：**依据——“It's also **a bit flimsy**, and I don't feel confident it will hold up for long **with heavy dishes.**”（结构单薄、承重场景下信心不足）
- **价值/对比劣势：**依据——“The **material feels a bit cheap for the price...** Overall, it's functional **but not quite what I expected given the cost.**”（价格与材料/体验不匹配，性价比不佳）

说明：关于“材质廉价→耐久性担忧”为感知与预期问题，未出现实际锈蚀/涂层脱落等失效证据，故不单独打“材质/防锈失效”，而归入“价值/对比劣势”。



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

I would give it more stars, except for the fact that apparently when parts didn't go on where they were supposed to from the manufacturer, they just throw it in the box. The little rubber things that are meant to support the end flaps we're not all on the rack the way they were supposed to be. There was one just thrown in the box. But there is absolutely no way to get it on the rack. It is something that has to be done during manufacture.

已思考 15s >

- **留言翻译：**

我本来想给更多星，但看起来厂家把没装好的零件就直接塞进盒子里了。用于支撑侧翼的小橡胶件并没有都装在架子上，其中一个是直接扔在盒子里的。而且根本装不上去——这必须在工厂装配完成。

- **舆情分析：**

明确负面影响为“到货即缺陷”，可用性受阻；用户指向制造/质检与包装流程问题，退货概率高，需立刻排查供应商装配工艺与出厂检验。

- **退货标签：原文依据**

- **损坏到货 (DOA/受损)**：依据——“**when parts didn't go on where they were supposed to from the manufacturer...**”、“**There was one just thrown in the box.**”（到货即为装配缺陷/缺件状态，影响正常使用）
- **安装/调节/固定性差**：依据——“**absolutely no way to get it on the rack. It has to be done during manufacture.**”（用户无法完成装配，需出厂预装，导致不可用）
- **包装防护不足**（从标签）：依据——“**just throw it in the box.**”（零件散落，包装/装箱未固定，增加缺失与损伤风险）



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

I noticed several problems. If you extend the rack for a larger sink, there are large gaps between the supports and items fall through to the sing below. The "cup" for silverware is very small and is too tall to be able to be linked to the side of the rack. A interesting idea but poorly implemented.

- **留言翻译：**

我发现了几个问题：把架子拉伸给更大的水槽用时，支撑条之间出现很大的缝隙，东西会从缝里掉到水槽里。放餐具的“杯筒”很小，而且太高，没法稳稳地挂在架子侧边。创意不错，但实现很差。

- **舆情分析：**

明确负面。核心为“扩展状态下有效承托不足”与“配件（刀叉杯）几何设计不当”，属于结构/设计层面的可用性问题；用户对实现质量失望，退换概率高。

- **退货标签：原文依据**

- **容量/有效面积不足（含扩容效果不达标）**：依据——“**If you extend the rack for a larger sink, there are large gaps between the supports and items fall through to the sink below.**”（拉伸后支撑间隙过大，无法有效承托物品）
- **分区/配件设计不合理（刀叉篮/杯架等）**：依据——“**The 'cup' for silverware is very small and is too tall to be able to be linked to the side of the rack.**”（刀叉杯过小且过高，无法侧挂/稳固）
- **价值/对比劣势（从标签）**：依据——“**An interesting idea but poorly implemented.**”（创意可以，但实现差，综合价值感偏低）



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

Arrived with the box already open and the rack missing one of the four knobs(D) for adjusting the sides, product still feels well made and sturdy enough with just the three knobs.

- **留言翻译：**

到货时盒子已经是开着的，而且用于调节侧边的四个旋钮（D）少了一个。虽然只用三个旋钮产品看起来也还算做工扎实、够稳。

- **舆情分析：**

中度负面（到货即问题）。核心在“开箱即缺件 + 包装不当/疑似二次开封”，虽用户认可一定稳固性，但完备性缺失触发退换动机与信任受损。

- **退货标签：原文依据**

- **包装防护不足：**依据——“Arrived with the box already open”（到货外箱已打开/封口失效，包装不当）。
- **损坏到货 (DOA/受损)：**依据——“missing one of the four knobs (D) for adjusting the sides”（开箱即缺件，影响完整功能与可调性）。



角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能会存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

This in-sink dish drying rack does the trick, really a great space saver for small kitchens with zero counter space. We don't have a dishwasher, so I have to hand wash literally everything and I do dishes at least once

a day.

We've had this drying rack for about 6 months and unfortunately, it does leave scratches on plates, bowls, mugs, and other dishes. When I feel around on the rack, I don't feel any sharp edges or anywhere that would cause the scratches, but my dishes still get dinged up. I get it, the dishes you use every day will get normal wear and tear, but these are our brand new dish sets and pots and pans that we got as wedding gifts, since we are newlyweds. Which is frustrating because I want to keep them nice so they last a long time. Oh well.

Perhaps we got a defective one, but just something to keep in mind when considering this dish rack.

已思考 18s >

- **留言翻译：**

这款槽内沥水架能用、也很省空间，适合没有台面的迷你厨房。我们没有洗碗机，所以每天都要手洗。用了大约 6 个月后，盘子、碗、杯子等餐具被刮出划痕。我摸过架子，没有明显尖锐处，但餐具还是会被磨花。虽然日常使用有磨损我能理解，但这些是我们新婚时的新餐具和锅具，我希望能保持如新、用得更久，所以很沮丧。也许我们这件是次品，但提醒大家购买时留意。

- **舆情分析：**

中度偏负的“长期使用痛点”。前半段肯定“省空间/功能可用”，但 6 个月后持续性“刮伤餐具”的负向体验权重更高，且与“珍�新婚餐具”情境绑定，易触发退换或差评。问题更像是表面处理/接触面保护不足导致的“隐性损伤”。

- **退货标签：原文依据**

- **台面/器具保护不足（防滑/防刮）**（主标签）：依据——“**it does leave scratches on plates, bowls, mugs, and other dishes.**”; “**I don't feel any sharp edges... but my dishes still get dinged up.**”（正常使用下接触面导致餐具被刮花/磨损）
- **(不打以下标签的说明)**
 - **材质/防锈失效**：文中无锈蚀/涂层脱落描述，仅为划伤餐具的问题，故不标。

- **场景适配不足/稳定性不足**: 用户认可“省空间/能用”，未提到倾倒/晃动/不稳，故不标。
- **价值/对比劣势**: 未出现价格/替代对比或换购倾向的明确信号，仅表达失望与可能“个别次品”猜测，暂不标。

□ ↗ ↙ ⌂ ⌃ ...

角色切换，现在你是一名亚马逊美国站点的运营小白，请你根据上述终版的退货标签以及定义边界，对下面这段退货留言进行打标，留言可能会涉及多个吐槽点，所以标签可能存在多个的情况，要点式展示（示例：留言翻译，舆情分析，退货标签：原文依据）：

This is a revised rating. Nice dish rack but it doesn't hold glasses well. I like the expandable feature but glasses fall over. Since we use this in the sink, our small plates go thru the slots and hit the sink.

已思考 34s >

- **留言翻译**:

这是更新后的评分。沥水架整体不错，但对玻璃杯的固定性很差。我喜欢可伸缩设计，但玻璃杯还是会倒。因为我们把它放在水槽里用，小盘子会从栅格缝里滑下去撞到水槽。

- **舆情分析**:

中度负面（功能点失效）。两大痛点：①玻璃杯易倒=杯架/分区设计不到位；②栅格间隙过大导致小盘承托不足。用户仍认可外观/概念，但实际可用性不足，存在降星/退换风险。

- **退货标签：原文依据**

- **分区/配件设计不合理（刀叉篮/杯架等）（主）**：依据——“**doesn't hold glasses well... glasses fall over.**”（杯具分区不能稳固杯子，属配件/分区设计问题而非整架普遍晃动）
- **容量/有效面积不足（含承托面/间隙过大）（主）**：依据——“**small plates go thru the slots and hit the sink.**”（栅格间隙过大，小盘无法被有效承托）
- **场景适配不足（槽内）（从）**：依据——“**Since we use this in the sink...**”（在“槽内”使用触发承托问题；根因是承托/间隙设计，场景为触发条件）



角色切换，现在你是一名数据分析师，任务是把终版的退货标签库写入Doris数据库，现在请你帮我设计表结构

下面给你一套面向 **Apache Doris** 的星型/多对多打标方案（支持一个留言多标签、主/从标签、版本管理）。包含核心表 + 支撑表，并附带 Doris 方言 DDL（含分区、分桶与索引）。你可以直接拷贝建表，再按集群调整 `replication_num / buckets`。

设计思路（超简）

- 维表：`dim_tag` 存放“终版退货标签库”（层级、定义、边界、版本）；`dim_tag_alias` 放别名/关键词；`dim_product` 放 ASIN。
- 事实表：`dwd_review` 存留言（原文/翻译等）；`fct_review_tag` 作为留言↔标签的桥表（多对多，含主/从、置信度、来源）。
- 可选：`dim_tag_example` 保存正反例句；`dim_tag_conflict` 维护互斥/优先级规则。

1) 退货标签维表（终版标签库）

sql

□ 复制代码

```
CREATE TABLE dim_tag (
    tag_id          BIGINT      COMMENT '标签ID',
    tag_code        VARCHAR(64) COMMENT '标签编码，如 FIT_COMPAT',
    tag_name_zh     VARCHAR(128) COMMENT '中文名',
    tag_name_en     VARCHAR(128) COMMENT '英文名（可选）',
    category_code   VARCHAR(64)  COMMENT '一级类目编码，如 CAT_FIT_SCENE',
    category_name_zh VARCHAR(128) COMMENT '一级类目中文名，如 适配与场景',
    parent_tag_id   BIGINT      COMMENT '父标签ID（顶层为NULL）',
    level           TINYINT     COMMENT '层级：1=一级类目，2=具体标签',
    definition      STRING      COMMENT '定义',
    boundary_note   STRING      COMMENT '与相近标签的边界说明',
    is_active       BOOLEAN     DEFAULT true,
    version         INT         DEFAULT 1 COMMENT '标签库版本',
    effective_from DATE        DEFAULT '2025-01-01',
    effective_to   DATE        COMMENT '失效日期, NULL=仍有效',
);
```

```

    created_at      DATETIME      DEFAULT CURRENT_TIMESTAMP,
    updated_at      DATETIME      DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
UNIQUE KEY(tag_id)
DISTRIBUTED BY HASH(tag_id) BUCKETS 8
PROPERTIES ("replication_num" = "3");

```

说明：把你“终版退货标签库”的每个一级类目作为 level=1、每个具体标签作为 level=2 插入； boundary_note 就写你整理的“边界划分”。

别名/关键词（用于检索、自动打标规则）

sql

复制代码

```

CREATE TABLE dim_tag_alias (
    tag_id      BIGINT,
    lang        VARCHAR(8) COMMENT 'zh/en 等',
    alias       VARCHAR(256) COMMENT '别名/关键词/同义表达',
    created_at  DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY(tag_id, lang, alias)
DISTRIBUTED BY HASH(tag_id) BUCKETS 8
PROPERTIES ("replication_num" = "3");

```

(可选) 标签互斥/优先级规则

sql

复制代码

```

CREATE TABLE dim_tag_conflict (
    tag_id          BIGINT,
    conflict_tag_id BIGINT,
    is_mutually_exclusive BOOLEAN DEFAULT true COMMENT '是否互斥',
    prefer_tag_id   BIGINT COMMENT '冲突时优先保留的tag_id（可选）',
    rule_note       STRING COMMENT '判定规则/说明',
    created_at      DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY(tag_id, conflict_tag_id)

```

```
DISTRIBUTED BY HASH(tag_id) BUCKETS 8
PROPERTIES ("replication_num" = "3");
```

(可选) 标签示例库 (正/反例)

sql

 复制代码

```
CREATE TABLE dim_tag_example (
    tag_id      BIGINT,
    lang        VARCHAR(8),
    example     STRING,
    is_positive BOOLEAN DEFAULT true COMMENT 'true=正例; false=反例/易混淆',
    source      VARCHAR(32) COMMENT 'human/model/real_review',
    created_at  DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY(tag_id, lang, example)
DISTRIBUTED BY HASH(tag_id) BUCKETS 8
PROPERTIES ("replication_num" = "3");
```

2) 商品维表 (ASIN)

sql

 复制代码

```
CREATE TABLE dim_product (
    asin          VARCHAR(20),
    product_title STRING,
    brand         VARCHAR(128),
    marketplace   VARCHAR(8) DEFAULT 'US',
    status         VARCHAR(32) DEFAULT 'ACTIVE',
    created_at    DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at    DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
UNIQUE KEY(asin)
DISTRIBUTED BY HASH(asin) BUCKETS 8
PROPERTIES ("replication_num" = "3");
```

3) 留言明细表 (可做 ODS/DWD)

支持全文检索 (INVERTED INDEX)，按天分区，便于增量加载与保留期管理。

sql

 复制代码

```

CREATE TABLE dwd_review (
    dt DATE COMMENT '评论日期或入库日期',
    review_id VARCHAR(64),
    asin VARCHAR(20),
    marketplace VARCHAR(8) DEFAULT 'US',
    rating TINYINT,
    title STRING,
    content_raw STRING COMMENT '原文',
    content_lang VARCHAR(8) COMMENT '原文语言',
    content_trans STRING COMMENT '机器/人工翻译（可选）',
    verified_purchase BOOLEAN,
    helpful_votes INT,
    reviewer_id VARCHAR(64),
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY(dt, review_id)
PARTITION BY RANGE(dt) ()
DISTRIBUTED BY HASH(review_id) BUCKETS 32
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-400",
    "dynamic_partition.end" = "7",
    "dynamic_partition.prefix" = "p",
    "dynamic_partition.buckets" = "32",
    "replication_num" = "3"
);

-- 全文索引，便于关键词/别名检索与回查
CREATE INDEX idx_content_raw_inverted ON dwd_review(content_raw) USING INVERTED;
CREATE INDEX idx_content_trans_inverted ON dwd_review(content_trans) USING INVERTED;

```

4) 留言-标签 事实表 (多对多桥表)

一条留言可有多标签；记录主/从标签、置信度、来源 (human/model) 、模型版本等。

sql

 复制代码

```

CREATE TABLE fct_review_tag (
    dt          DATE,
    review_id   VARCHAR(64),
    asin        VARCHAR(20),
    tag_id      BIGINT,
    tag_role    VARCHAR(16) COMMENT 'primary/secondary',
    confidence  DECIMAL(5,4) COMMENT '0~1',
    source      VARCHAR(16) COMMENT 'human/model/rule',
    annotator   VARCHAR(64) COMMENT '标注人/系统',
    model_version VARCHAR(64) COMMENT '模型或规则版本',
    created_at   DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY(dt, review_id, tag_id)
PARTITION BY RANGE(dt) ()
DISTRIBUTED BY HASH(review_id) BUCKETS 32
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-400",
    "dynamic_partition.end" = "7",
    "dynamic_partition.prefix" = "p",
    "dynamic_partition.buckets" = "32",
    "replication_num" = "3"
);

-- 常用查询加速
CREATE INDEX idx_tag_bitmap ON fct_review_tag(tag_id) USING BITMAP;
CREATE INDEX idx_role_bitmap ON fct_review_tag(tag_role) USING BITMAP;

```

5) 建议的编码规范 (示例)

- category_code (一级类目) :
 - CAT_FIT_SCENE (适配与场景)
 - CAT_STRUCT_LOAD (结构与承载)
 - CAT_MATERIAL_LOOK (材质与外观)
 - CAT_INSTALL_USE (安装与使用体验)
 - CAT_INFO_EXPECT (信息与预期管理)
 - CAT_PRICE_POLICY (价格与政策)
 - CAT_FULFILL_SERVICE (履约与服务)
- tag_code (二级标签示例) :
 - FIT_COMPAT, SCENE_ADAPT, GUIDE_VARIANT,
 - STABILITY_TIP, CAPACITY_EFF, PARTS DESIGN, WEIGHT_MOVE,
 - MATERIAL_RUST, LOOK_COLOR, SURFACE_PROTECT, CLEAN_MAINT,
 - INSTALL_FIX,
 - SPEC_MISMATCH, PAGE_MISSING, SOCIAL_GAP,
 - PRICE_PROMO_OPAQ, VALUE_WEAK, RETURN_POLICY_BAD,
 - SLA_DELAY, PACK_WEAK, DOA_DAMAGED, CS_POOR.

这样你可以把“终版标签库”的每条**一级/二级**都落到 dim_tag，并用 parent_tag_id 建层级，保证 **MECE** 与可扩展。

6) 常见查询示例 (验证链路)

sql

 复制代码

```
-- 最近30天，按二级标签统计被打标的留言数
SELECT t.tag_name_zh, COUNT(*) AS cnt
FROM fct_review_tag f
JOIN dim_tag t ON f.tag_id = t.tag_id
WHERE f.dt >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
  AND t.level = 2
GROUP BY t.tag_name_zh
ORDER BY cnt DESC;
```

```
-- 查看某ASIN下“稳定性/防倾倒不足”的典型原文
SELECT r.review_id, r.content_raw
FROM fct_review_tag f
```

```

JOIN dwd_review r ON f.review_id = r.review_id AND f.dt = r.dt
JOIN dim_tag t ON f.tag_id = t.tag_id
WHERE r.asin = 'B0BGHGXYJX' AND t.tag_code = 'STABILITY_TIP'
ORDER BY r.dt DESC
LIMIT 50;

```

如果你愿意，我可以把你“终版标签库”的**每一条**转换成 `INSERT INTO dim_tag(...)` `VALUES (...)` 的模板文件，直接批量落库。



↳ 如果你愿意，我可以把你“终版标签库”的**每一条**转换成 `INSERT INTO dim_tag(...)` `VALUES (...)` 的模板文件，直接批量落库。

把“终版标签库”的**每一条**转换成 `INSERT INTO dim_tag(...)`
`VALUES (...)`

下面给出把“终版退货标签库”逐条落库到 `dim_tag` 的 SQL (含**一级类目与二级标签**)。字段按你之前的表结构：

```
(tag_id, tag_code, tag_name_zh, tag_name_en, category_code, category_name_zh,
parent_tag_id, level, definition, boundary_note, is_active, version,
effective_from, effective_to)
```

说明：

- “**一级类目** `parent_tag_id = NULL`, `level = 1`；**二级标签** `parent_tag_id =` 对应一级类目的 `tag_id`, `level = 2`。”
- “可按需调整 `tag_id/tag_code/英文名/生效日期`。”
- “若你已在库中占用了这些 `tag_id`，可整体平移或改为自增。”

sql

复制代码

```
-- =====
-- 一级类目 (Level 1)
-- =====
INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(1000, 'CAT_FIT_SCENE', '适配与场景', 'Fit & Scenarios', 'CAT_FIT_SCENE', '适配与场景', N
```

```
(1001, 'CAT_STRUCT_LOAD', '结构与承载', 'Structure & Load', 'CAT_STRUCT_LOAD', '结构与承
(1002, 'CAT_MATERIAL_LOOK', '材质与外观', 'Material & Appearance', 'CAT_MATERIAL_LOOK'
(1003, 'CAT_INSTALL_USE', '安装与使用体验', 'Installation & Use', 'CAT_INSTALL_USE', '安
(1004, 'CAT_INFO_EXPECT', '信息与预期管理', 'Info & Expectation Mgmt', 'CAT_INFO_EXPECT'
(1005, 'CAT_PRICE_POLICY', '价格与政策', 'Price & Policy', 'CAT_PRICE_POLICY', '价格与政
(1006, 'CAT_FULFILL_SERVICE', '履约与服务', 'Fulfillment & Service', 'CAT_FULFILL_SERV
```

-- =====

-- 二级标签 (Level 2) —适配与场景

-- =====

```
INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2000, 'FIT_COMPAT', '尺寸/兼容性不符', 'Size/Fit Mismatch', 'CAT_FIT_SCENE', '适配与场
(2001, 'SCENE_ADAPT', '场景适配不足（槽内/跨槽/台面/RV 等）', 'Scenario Suitability Iss
(2002, 'GUIDE_VARIANT', '变体/选型指引不清', 'Variant/Selection Guidance Unclear', 'CA
```

-- =====

-- 二级标签 (Level 2) —结构与承载

-- =====

```
INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2100, 'STABILITY_TIP', '稳定性/防倾倒不足', 'Stability / Anti-Tip Insufficient', 'CAT_
(2101, 'CAPACITY_EFF', '容量/有效面积不足（含扩容效果不达标）', 'Capacity / Effective Ar
(2102, 'PARTS_DESIGN', '分区/配件设计不合理（刀叉篮/杯架等）', 'Partition/Accessory Des
(2103, 'WEIGHT_MOVE', '自重过大/移动不便', 'Too Heavy / Hard to Move', 'CAT_STRUCT_LOAD
```

-- =====

-- 二级标签 (Level 2) —材质与外观

-- =====

```
INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2200, 'MATERIAL_RUST', '材质/防锈失效', 'Material / Anti-rust Failure', 'CAT_MATERIAL_
(2201, 'LOOK_COLOR', '外观/颜色与实物不符', 'Appearance / Color Mismatch', 'CAT_MATERIAL
(2202, 'SURFACE_PROTECT', '台面/器具保护不足（防滑/防刮）', 'Surface Protection Insuffi
(2203, 'CLEAN_MAINT', '清洁/维护成本高', 'Cleaning / Maintenance Burden', 'CAT_MATERIAL
```

-- =====

-- 二级标签 (Level 2) —安装与使用体验

-- =====

```
INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2300, 'INSTALL_FIX', '安装/调节/固定性差', 'Install / Adjustment / Fixation Issues', '
```

-- =====

```
-- 二级标签 (Level 2) — 信息与预期管理
-- =====

INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2400, 'SPEC_MISMATCH', '规格参数与宣传不一致', 'Specs vs Listing Inconsistency', 'CAT_
(2401, 'PAGE_MISSING', '页面信息缺失/误导', 'Listing Info Missing / Misleading', 'CAT_]
(2402, 'SOCIAL_GAP', '社会证明与体验不符', 'Social Proof vs Experience Gap', 'CAT_INFO_


-- =====
-- 二级标签 (Level 2) — 价格与政策
-- =====

INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2500, 'PRICE_PROMO_OPAQ', '价格/促销不透明', 'Pricing / Promotion Opacity', 'CAT_PRICE_
(2501, 'VALUE_WEAK', '价值/对比劣势', 'Value Weak vs Alternatives', 'CAT_PRICE_POLICY'
(2502, 'RETURN_POLICY_BAD', '退换信息/政策体验不佳', 'Return Policy Experience Poor', '


-- =====
-- 二级标签 (Level 2) — 履约与服务
-- =====

INSERT INTO dim_tag (tag_id, tag_code, tag_name_zh, tag_name_en, category_code, c
(2600, 'SLA_DELAY', '物流时效异常', 'Delivery SLA Delay', 'CAT_FULFILL_SERVICE', '履约与
(2601, 'PACK_WEAK', '包装防护不足', 'Packaging Protection Weak', 'CAT_FULFILL_SERVICE'
(2602, 'DOA_DAMAGED', '损坏到货 (DOA/受损)', 'Damaged on Arrival', 'CAT_FULFILL_SERVIC
(2603, 'CS_POOR', '品牌/店铺服务不达标', 'Brand/Store Support Poor', 'CAT_FULFILL_SERVI
```

如果需要，我可以把上述 SQL 另存为 .sql 文件并附带一个 INSERT dim_tag_alias 的模板（中英文关键词/常见同义表达），方便你直接在 Doris 中执行。



dim_tag的表结构改成下面这样子，请你对应给出落库到
return_dim_tag 的 SQL

```
CREATE TABLE hyy.return_dim_tag (
    tag_code      VARCHAR(64) COMMENT '标签编码',
    tag_name_cn   VARCHAR(128) COMMENT '中文名',
    category_code VARCHAR(64) COMMENT '一级类目
        编码',
    category_name_cn VARCHAR(128) COMMENT '一级
        类目中文名',
```

```

level      TINYINT   COMMENT '层级: 1=一级类
目, 2=具体标签',
definition  STRING    COMMENT '标签定义',
boundary_note  STRING    COMMENT '与相近标签
的边界说明',
is_active    BOOLEAN   DEFAULT 1 COMMENT '是
否启用',
version      INT      DEFAULT 1 COMMENT '标签库
版本',
effective_from DATE    DEFAULT '2025-11-01',
effective_to   DATE    COMMENT '失效日期,
NULL=仍有效',
created_at    DATETIME  DEFAULT
CURRENT_TIMESTAMP,
updated_at    DATETIME  DEFAULT
CURRENT_TIMESTAMP
)
ENGINE=OLAP
UNIQUE KEY(tag_code)
COMMENT '退货标签维表: 适用于退货留言/买家之声/页
面评论'
DISTRIBUTED BY HASH(tag_code) BUCKETS 12
PROPERTIES (
"replication_allocation" = "tag.location.default: 3",
"min_load_replica_num" = "-1",
"is_being_synced" = "false",
"storage_medium" = "hdd",
"storage_format" = "V2",
"inverted_index_storage_format" = "V1",
"light_schema_change" = "true",
"disable_auto_compaction" = "false",
"enable_single_replica_compaction" = "false",
"group_commit_interval_ms" = "10000",
"group_commit_data_bytes" = "134217728"
);

```

下面给出按你提供的新表结构 **hyy.return_dim_tag** 的落库 SQL。已去掉 `tag_id` 等字
段，分别插入**一级类目 (level=1)** 与**二级标签 (level=2)**；`created_at/updated_at` 使
用默认值，`effective_to` 统一为 `NULL`。如需批量执行，可整段复制运行。

sql

复制代码

```
-- =====
-- Level 1: 一级类目
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('CAT_FIT_SCENE', '适配与场景', 'CAT_FIT_SCENE', '适配与场景', 1, '与水槽/安装环境相关的适配', 1),
('CAT_STRUCT_LOAD', '结构与承载', 'CAT_STRUCT_LOAD', '结构与承载', 1, '结构稳定、承重、容积', 1),
('CAT_MATERIAL_LOOK', '材质与外观', 'CAT_MATERIAL_LOOK', '材质与外观', 1, '材质耐久、外观美观', 1),
('CAT_INSTALL_USE', '安装与使用体验', 'CAT_INSTALL_USE', '安装与使用体验', 1, '装配、调节、安装便利性', 1),
('CAT_INFO_EXPECT', '信息与预期管理', 'CAT_INFO_EXPECT', '信息与预期管理', 1, '规格准确性、信息透明度', 1),
('CAT_PRICE_POLICY', '价格与政策', 'CAT_PRICE_POLICY', '价格与政策', 1, '价格透明、价值感知', 1),
('CAT_FULFILL_SERVICE', '履约与服务', 'CAT_FULFILL_SERVICE', '履约与服务', 1, '物流时效、售后服务', 1);

-- =====
-- Level 2: 适配与场景
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('FIT_COMPAT', '尺寸/兼容性不符', 'CAT_FIT_SCENE', '适配与场景', 2, '在标称适配范围内仍无法满足', 2),
('SCENE_ADAPT', '场景适配不足（槽内/跨槽/台面/RV 等）', 'CAT_FIT_SCENE', '适配与场景', 2, '场景适应性不足', 2),
('GUIDE_VARIANT', '变体/选型指引不清', 'CAT_FIT_SCENE', '适配与场景', 2, '尺寸/版本差异与适应性', 2);

-- =====
-- Level 2: 结构与承载
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('STABILITY_TIP', '稳定性/防倾倒不足', 'CAT_STRUCT_LOAD', '结构与承载', 2, '出现下弯、晃动', 2),
('CAPACITY_EFF', '容量/有效面积不足（含扩容效果不达标）', 'CAT_STRUCT_LOAD', '结构与承载', 2, '容量不足', 2),
('PARTS DESIGN', '分区/配件设计不合理（刀叉篮/杯架等）', 'CAT_STRUCT_LOAD', '结构与承载', 2, '分区设计不合理', 2),
('WEIGHT_MOVE', '自重过大/移动不便', 'CAT_STRUCT_LOAD', '结构与承载', 2, '整机偏重或抓握不稳', 2);

-- =====
-- Level 2: 材质与外观
-- =====
```

```
INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('MATERIAL_RUST', '材质/防锈失效', 'CAT_MATERIAL_LOOK', '材质与外观', 2, '短期即出现锈斑、',
('LOOK_COLOR', '外观/颜色与实物不符', 'CAT_MATERIAL_LOOK', '材质与外观', 2, '与商品图或宣和',
('SURFACE_PROTECT', '台面/器具保护不足（防滑/防刮）', 'CAT_MATERIAL_LOOK', '材质与外观', 2,
('CLEAN_MAINT', '清洁/维护成本高', 'CAT_MATERIAL_LOOK', '材质与外观', 2, '易积污、死角多、

-- =====
-- Level 2: 安装与使用体验
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('INSTALL_FIX', '安装/调节/固定性差', 'CAT_INSTALL_USE', '安装与使用体验', 2, '装配虽易，但

-- =====
-- Level 2: 信息与预期管理
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('SPEC_MISMATCH', '规格参数与宣传不一致', 'CAT_INFO_EXPECT', '信息与预期管理', 2, '伸缩范围不
('PAGE_MISSING', '页面信息缺失/误导', 'CAT_INFO_EXPECT', '信息与预期管理', 2, '缺关键尺寸图
('SOCIAL_GAP', '社会证明与体验不符', 'CAT_INFO_EXPECT', '信息与预期管理', 2, '热卖/高评等社

-- =====
-- Level 2: 价格与政策
-- =====

INSERT INTO hyy.return_dim_tag
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('PRICE_PROMO_OPAQ', '价格/促销不透明', 'CAT_PRICE_POLICY', '价格与政策', 2, '优惠券未应用
('VALUE_WEAK', '价值/对比劣势', 'CAT_PRICE_POLICY', '价格与政策', 2, '与同价替代品相比综合
('RETURN_POLICY_BAD', '退换信息/政策体验不佳', 'CAT_PRICE_POLICY', '价格与政策', 2, '退换

-- =====
-- Level 2: 履约与服务
-- =====

INSERT INTO hyy.return_dim_tag
```

```
(tag_code, tag_name_cn, category_code, category_name_cn, level, definition, bound)
VALUES
('SLA_DELAY', '物流时效异常', 'CAT_FULFILL_SERVICE', '履约与服务', 2, '承诺时效未达, 延迟至',
('PACK_WEAK', '包装防护不足', 'CAT_FULFILL_SERVICE', '履约与服务', 2, '外箱/内衬防护不足,
('DOA_DAMAGE', '损坏到货 (DOA/受损)', 'CAT_FULFILL_SERVICE', '履约与服务', 2, '开箱即明
('CS_POOR', '品牌/店铺服务不达标', 'CAT_FULFILL_SERVICE', '履约与服务', 2, '客服响应慢、处
```

如需把这些语句导出为 .sql 文件、或追加英文名/多语言别名表（用于自动打标关键词），我可以一并生成。



留言-标签 事实表将会包括下面的内容，如何设计表结构呢？

****review_id**:** 702-9563280-0645818_FPL1001M

****留言来源**:** 退货留言/买家之声/页面评论

****留言原文**:**

This is a revised rating. Nice dish rack but it doesn't hold glasses well. I like the expandable feature but glasses fall over. Since we use this in the sink, our small plates go thru the slots and hit the sink.

* ****留言翻译**:**

这是更新后的评分。沥水架整体不错，但对玻璃杯的固定性很差。我喜欢可伸缩设计，但玻璃杯还是会倒。因为我们把它放在水槽里用，小盘子会从栅格缝里滑下去撞到水槽。

* ****舆情分析**:**

中度负面（功能点失效）。两大痛点：①玻璃杯易倒=杯架/分区设计不到位；②栅格间隙过大导致小盘承托不足。

用户仍认可外观/概念，但实际可用性不足，存在降星/退换风险。

* ****退货标签：原文依据****

* ****分区/配件设计不合理（刀叉篮/杯架等）** (主) :**
依据——“****doesn't hold glasses well** ... **glasses fall over**.**” (杯具分区不能稳固杯子，属配件/分区设计问题而非整架普遍晃动)

* **容量/有效面积不足（含承托面/间隙过大）**
 (主)：依据——“small plates **go thru the slots**”
 and **hit the sink**.” (栅格间隙过大，小盘无法被有效承
 托)
 * **场景适配不足（槽内）** (从)：依据——“**Since
 we use this in the sink**...” (在“槽内”使用触发承托问
 题；根因是承托/间隙设计，场景为触发条件)

下面给你两种在 **Apache Doris** 上可落地的设计方案，便于同时保存“留言内容 + 情绪分析 + 多标签及证据”。推荐用**规范化（两张表）**，也提供**单表 JSON** 方案以便快速落地。

方案 A (推荐) : 规范化两表 + 视图

一条留言可关联多条标签记录；每条标签可带“主/从、证据片段、置信度、来源（人工/模型）”。

1) 留言事实表 (含原文/翻译/情绪)

sql

复制代码

```
CREATE TABLE hyy.return_review_fact (
    dt DATE COMMENT '业务日期/入库日',
    review_id VARCHAR(128) COMMENT '如: 702-956328',
    asin VARCHAR(20) COMMENT '可选: 所属ASIN',
    source_type VARCHAR(32) COMMENT '留言来源: return',
    rating TINYINT COMMENT '若为评论场景可选',
    content_raw STRING COMMENT '留言原文',
    content_lang VARCHAR(8) DEFAULT 'en' COMMENT '原文语言',
    content_trans STRING COMMENT '留言翻译 (中文)',
    sentiment_label VARCHAR(16) COMMENT 'neg/neu/pos 等',
    sentiment_score DECIMAL(5,4) COMMENT '0~1 置信度',
    analysis_summary STRING COMMENT '舆情要点/分析摘要',
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id)
PARTITION BY RANGE (dt) ()
```

```
DISTRIBUTED BY HASH (review_id) BUCKETS 16
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-180",
    "dynamic_partition.end" = "7",
    "dynamic_partition.prefix" = "p",
    "dynamic_partition.buckets" = "16",
    "replication_allocation" = "tag.location.default: 3"
);
```

-- 文本倒排索引，便于关键句/证据回查

```
CREATE INDEX idx_rrf_raw_inverted ON hyy.return_review_fact(content_raw) USING
CREATE INDEX idx_rrf_tran_inverted ON hyy.return_review_fact(content_trans) USING
```

2) 留言-标签事实表 (桥表：一条留言↔多标签)

sql

 复制代码

```
CREATE TABLE hyy.return_review_tag_fact (
    dt DATE,
    review_id VARCHAR(128),
    tag_code VARCHAR(64) COMMENT '参照 hyy.return_dim',
    tag_role VARCHAR(16) COMMENT 'primary/secondary',
    evidence_text STRING COMMENT '原文依据片段',
    evidence_start INT COMMENT '片段在原文的起始字符',
    evidence_end INT COMMENT '片段在原文的结束字符',
    confidence DECIMAL(5,4) COMMENT '0~1 置信度',
    source VARCHAR(16) COMMENT 'human/model/rule',
    model_version VARCHAR(64) COMMENT '模型或规则版本 (人工',
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id, tag_code, tag_role)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (review_id) BUCKETS 16
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-180",
```

```

"dynamic_partition.end" = "7",
"dynamic_partition.prefix" = "p",
"dynamic_partition.buckets" = "16",
"replication_allocation" = "tag.location.default: 3"
);

```

-- 标签与角色做位图索引，常见聚合更快

```

CREATE INDEX idx_rrtf_tag_bitmap ON hyy.return_review_tag_fact(tag_code) USING B
CREATE INDEX idx_rrtf_role_bitmap ON hyy.return_review_tag_fact(tag_role) USING B

```

3) (可选) 便捷查询视图：留言 + 标签展开

sql

 复制代码

```

CREATE VIEW hyy.v_return_review_with_tags AS
SELECT
    r.dt, r.review_id, r.asin, r.source_type, r.rating,
    r.content_raw, r.content_trans,
    r.sentiment_label, r.sentiment_score, r.analysis_summary,
    t.tag_code, d.tag_name_cn, d.category_name_cn, t.tag_role,
    t.evidence_text, t.confidence, t.source, t.model_version
FROM hyy.return_review_fact r
LEFT JOIN hyy.return_review_tag_fact t
    ON r.dt = t.dt AND r.review_id = t.review_id
LEFT JOIN hyy.return_dim_tag d
    ON t.tag_code = d.tag_code;

```

(按你给的示例) 落一条数据的样例

sql

 复制代码

-- 1) 留言主体

```

INSERT INTO hyy.return_review_fact
(dt, review_id, asin, source_type, rating, content_raw, content_lang, content_trans,
sentiment_label, sentiment_score, analysis_summary)
VALUES
(CURDATE(), '702-9563280-0645818_FPL1001M', NULL, 'review', NULL,
'This is a revised rating. Nice dish rack but it doesn''t hold glasses well. I like
'en',

```

'这是更新后的评分。沥水架整体不错，但对玻璃杯的固定性很差.....小盘子会从栅格缝里滑下去撞到水

```
'neg', 0.80,
'功能点失效：杯具易倒、槽内场景小盘下滑');

-- 2) 关联标签（主、主、从）
INSERT INTO hyy.return_review_tag_fact
(dt, review_id, tag_code, tag_role, evidence_text, confidence, source, model_version)
VALUES
(CURDATE(), '702-9563280-0645818_FPL1001M', 'PARTS_DESIGN', 'primary', 'doesn't',
(CURDATE(), '702-9563280-0645818_FPL1001M', 'CAPACITY_EFF', 'primary', 'small',
(CURDATE(), '702-9563280-0645818_FPL1001M', 'SCENE_ADAPT', 'secondary', 'Since 1
```

优点

- 满足一对多标签；同一标签可多次证据（可用 tag_role + 起止位定位）。
- 查询灵活，和 return_dim_tag 自然关联；分区/哈希利于增量装载与大规模聚合。

方案 B：单表存储 (JSON 列承载多标签)

适合快速落地、数据量中小或以检索为主的场景。多标签以 JSON 数组保存。

sql

 复制代码

```
CREATE TABLE hyy.return_review_fact_json (
    dt DATE,
    review_id VARCHAR(128),
    asin VARCHAR(20),
    source_type VARCHAR(32),
    rating TINYINT,
    content_raw STRING,
    content_lang VARCHAR(8) DEFAULT 'en',
    content_trans STRING,
    sentiment_label VARCHAR(16),
    sentiment_score DECIMAL(5,4),
    analysis_summary STRING,
    tags_json JSON COMMENT '示例: [{"tag_code": "PARTS_DESIGN", "role": "primary", "text": "杯具易倒"}, {"tag_code": "CAPACITY_EFF", "role": "primary", "text": "槽内场景小盘下滑"}]',
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id)
```

```

PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (review_id) BUCKETS 16
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-180",
    "dynamic_partition.end" = "7",
    "dynamic_partition.prefix" = "p",
    "dynamic_partition.buckets" = "16",
    "replication_allocation" = "tag.location.default: 3"
);

```

-- 可对 `content_raw/content_trans` 建倒排索引；也可对 `JSON` 提取虚拟列做查询（`Doris` 支持 `CREATE INDEX idx_rrfj_raw_inverted ON hyy.return_review_fact_json(content_raw)`）

权衡

- 单表写入简单，但多标签的聚合（如“按 `tag_code` 统计”）需要 `json_each / json_extract` 展开，性能略逊于方案 A。
- 若后续要做“证据级”分析或 A/B 多版本模型打标，建议迁移至**方案 A**。

如果你要把已有的“打标输出（CSV/Parquet）”批量导入 Doris，我也可以给你准备 **Stream Load / Broker Load** 的模板，以及基于方案 A 的 **Upsert (MERGE ON)** 示例。



方案B给一条数据示例看看

下面是一条基于 **方案 B (JSON 结构)** 的示例数据插入语句，直接可执行在 Doris 上的表 `hyy.return_review_fact_json` 中。

示例 SQL

```
INSERT INTO hyy.return_review_fact_json
(dt, review_id, asin, source_type, rating, content_raw, content_lang,
content_trans, sentiment_label, sentiment_score, analysis_summary, tags_json)
VALUES
(
CURDATE(),
'702-9563280-0645818_FPL1001M',
'B0D6YNTRTZ',
'review',
NULL,
'This is a revised rating. Nice dish rack but it doesn''t hold glasses well. I
'en',
'这是更新后的评分。沥水架整体不错，但对玻璃杯的固定性很差。我喜欢可伸缩设计，但玻璃杯还
'neg',
0.80,
'功能点失效：杯具易倒、槽内场景小盘下滑。',
JSON_PARSE([
{
    "tag_code": "PARTS_DESIGN",
    "tag_name_cn": "分区/配件设计不合理（刀叉篮/杯架等）",
    "role": "primary",
    "evidence": "doesn''t hold glasses well; glasses fall over",
    "confidence": 0.90,
    "source": "human"
},
{
    "tag_code": "CAPACITY_EFF",
    "tag_name_cn": "容量/有效面积不足（含承托面/间隙过大）",
    "role": "primary",
    "evidence": "small plates go thru the slots and hit the sink",
    "confidence": 0.85,
    "source": "human"
},
{
    "tag_code": "SCENE_ADAPT",
    "tag_name_cn": "场景适配不足（槽内/跨槽/台面/RV 等）",
    "role": "secondary",
    "evidence": "Since we use this in the sink",
    "confidence": 0.70,
}
```

```

    "source": "human"
}
]')
);

```

说明

字段	含义	示例值
dt	数据入库日期	CURDATE()
review_id	唯一留言编号	702-9563280-0645818_FPL1001M
asin	产品 ASIN	B0D6YNTRTZ
source_type	留言来源	review (可取 return、voc 等)
content_raw / content_trans	原文与翻译	英文与中文文本
sentiment_label / sentiment_score	情感标签与置信度	neg , 0.80
analysis_summary	舆情摘要	"功能点失效：杯具易倒、槽内场景小盘下滑。"
tags_json	JSON 数组，存储多标签及证据	三个标签对象：PARTS DESIGN 、 CAPACITY_EFF 、 SCENE_ADAPT

Doris 中查询示例

sql

 复制代码

```

-- 提取所有标签的tag_code
SELECT review_id, json_each.tags_json.tag_code
FROM hyy.return_review_fact_json, LATERAL VIEW json_each(tags_json) AS json_each;

```

```
-- 按标签统计评论数
SELECT json_extract_scalar(tag.value, '$.tag_code') AS tag_code,
       COUNT(*) AS cnt
FROM hyy.return_review_fact_json,
     LATERAL VIEW json_each(tags_json) tag
GROUP BY tag_code;
```

此示例可直接导入 Doris，便于快速落地测试，也能兼容后续批量 Stream Load 导入。



留言打标会通过调用deepseek的api，llm的方式实现，所以要尽量减少输入输出的token数量，省钱，优先用json格式输出入库，请你重新思考入库格式

很赞的目标！为“用 DeepSeek 打标、尽量少 token、直接入库”为核心，我给你一套**极简 JSON 出入库格式 + Doris 表结构**（单表落地，后续再 ETL 到规范化模型）。思路：**短键名、枚举数值化、复用编码 (tag_code) 、只保留必要字段**。

一、极简 JSON (LLM 输出/入库统一格式 v1)

约 30–60% token 节省；所有键名短化，枚举数字化；只保留最小可用信息。

json

复制代码

```
{
  "rid": "702-9563280-0645818_FPL1001M",           // review_id
  "asn": "B0D6YNTRZ",                                // ASIN (可选，无则省略)
  "src": 2,                                         // 留言来源: 0=return, 1=voc, 2=review
  "lang": "en",                                      // 原文语言 (无变化可省略)
  "raw": "This is a revised rating ...",            // 原文
  "zh": "这是更新后的评分 ...",                      // 翻译 (如无翻译可省略)
  "sen": -1,                                         // 情感标签: -1=neg, 0=neu, 1=pos
  "sc": 0.80,                                       // 情感置信度(0~1, 可选)
  "tags": [
```

```

    {"tc":"PARTS_DESIGN","r":1,"ev":"doesn't hold glasses well; glasses fall over
    {"tc":"CAPACITY_EFF","r":1,"ev":"small plates go thru the slots and hit the s
    {"tc":"SCENE_ADAPT","r":0,"ev":"Since we use this in the sink","cf":0.70}
]
}

```

字段说明 (尽量短) :

- rid (必) 评论/留言唯一 ID
- asn (选) ASIN
- src (必) 来源: 0=return (退货留言) /1=voc (买家之声) /2=review (页面评论)
- lang (选) 原文语言
- raw (必) 原文
- zh (选) 中文翻译
- sen (必) 情感: -1/0/1
- sc (选) 情感置信度
- tags (必) 标签数组; 元素:
 - tc 标签编码 (对应 return_dim_tag.tag_code)
 - r 主从: 1=primary,0=secondary
 - ev 证据片段 (原文)
 - cf 置信度 (0~1)

可进一步省钱: 没有翻译就不返回 zh , 没有情感模型就不返回 sc ; 证据很长可裁成短句。

二、Doris 单表落地 (方案 B 的更轻量版本)

一张表即可直接承接 LLM 输出; 后续用 SQL 将 JSON 展开, 灌入规范化桥表。

sql

复制代码

```

CREATE TABLE hyy.return_review_fact_min (
  dt          DATE                COMMENT '入库日期',
  rid         VARCHAR(128)        COMMENT 'review_id',
  asn         VARCHAR(20)         COMMENT 'ASIN',
  src         TINYINT            COMMENT '0=return,1=voc,2=review',
  lang        VARCHAR(8)          COMMENT '原文语言',

```

```

raw      STRING          COMMENT '原文',
zh       STRING          COMMENT '中文翻译(可空)',
sen      TINYINT         COMMENT '-1/0/1',
sc       DECIMAL(3,2)     COMMENT '情感分(0-1, 可空)',
tags    JSON             COMMENT '极简标签数组: [{tc,r,ev,...}],
created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, rid)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (rid) BUCKETS 16
PROPERTIES (
  "dynamic_partition.enable" = "true",
  "dynamic_partition.time_unit" = "DAY",
  "dynamic_partition.start" = "-90",
  "dynamic_partition.end" = "7",
  "dynamic_partition.prefix" = "p",
  "dynamic_partition.buckets" = "16",
  "replication_allocation" = "tag.location.default: 3"
);
-- 倒排索引 (可选, 利于检索证据/原文):
CREATE INDEX idx_rrfmin_raw_inv ON hyy.return_review_fact_min(raw) USING INVERTED

```

三、最小入库示例 (INSERT + JSON_PARSE)

sql

□ 复制代码

```

INSERT INTO hyy.return_review_fact_min
(dt, rid, asn, src, lang, raw, zh, sen, sc, tags)
VALUES
(
  CURDATE(),
  '702-9563280-0645818_FPL1001M',
  'B0D6YNTRTZ',
  2,
  'en',
  'This is a revised rating. Nice dish rack but it doesn''t hold glasses well. I

```

```
'这是更新后的评分。沥水架整体不错，但对玻璃杯的固定性很差。我喜欢可伸缩设计，但玻璃杯还
-1,
0.80,
JSON_PARSE('[
    {"tc": "PARTS_DESIGN", "r": 1, "ev": "doesn't hold glasses well; glasses fall ove
    {"tc": "CAPACITY_EFF", "r": 1, "ev": "small plates go thru the slots and hit the s
    {"tc": "SCENE_ADAPT", "r": 0, "ev": "Since we use this in the sink", "cf": 0.70}
]')
);
```

四、从轻量表展开为分析视图 / 规范化桥表

临时视图（查询时展开 tags）：

sql

 复制代码

```
SELECT
    dt, rid, asn, src, lang, raw, sen, sc,
    json_extract_scalar(t.value, '$.tc') AS tag_code,
    CAST(json_extract_scalar(t.value, '$.r') AS TINYINT) AS tag_role,
    json_extract_scalar(t.value, '$.ev') AS evidence,
    CAST(json_extract_scalar(t.value, '$.cf') AS DECIMAL(3,2)) AS confidence
FROM hyy.return_review_fact_min
LATERAL VIEW json_each(tags) t;
```

批量灌入规范化桥表（可按天跑）：

sql

 复制代码

```
INSERT INTO hyy.return_review_tag_fact
(dt, review_id, tag_code, tag_role, evidence_text, confidence, source, model_version)
SELECT
    a.dt, a.rid,
    json_extract_scalar(t.value, '$.tc') AS tag_code,
    IF(CAST(json_extract_scalar(t.value, '$.r') AS TINYINT)=1, 'primary', 'secondary'),
    json_extract_scalar(t.value, '$.ev') AS evidence_text,
    CAST(json_extract_scalar(t.value, '$.cf') AS DECIMAL(5,4)) AS confidence,
    'model' AS source,
    'deepseek-v1' AS model_version
```

```
FROM hyy.return_review_fact_min a
LATERAL VIEW json_each(a.tags) t
WHERE a.dt = CURDATE();
```

五、落模提示（进一步省钱）

- 短键名即规范**: 锁定上面的键名，Prompt 中明示“仅输出这些键，禁止多余文本”。
- 可选字段省略**: 没有翻译就不返回 zh；没有情感就不返回 sc；**不要输出空字符串**。
- 证据裁剪**: ev 只留触发短语，不贴整段（DeepSeek 输出也更稳）。
- 压缩**: Stream Load 开启 compress_type = gzip。
- 错误兜底**: 服务端做 JSON schema 校验（键缺失/类型错直接拒收），保持数据质量。

如果你给我你现有的 DeepSeek 调用 Prompt，我可以帮你把“输出规范化指令”再缩到极致（英文短句+示例），进一步减少 token。



极简 JSON (LLM 输出/入库统一格式 v1) 改成下面这样，然后如何入库呢？是直接 JSON 文件入库还是创建表入库呢？

```
{
    "review_id": "702-9563280-0645818_FPL1001M",
    "review_source": 2, // 留言来源: 0=return,1=voc,2=review
    "review_en": "This is a revised rating ...", // 留言原文
    "review_cn": "这是更新后的评分 ...", // 留言翻译
    "sentiment": -1, // 情感标签: -1=neg,0=neu,1=pos
    "tags": [
        {"tc": "PARTS_DESIGN", "ev": "doesn't hold glasses well; glasses fall over"},
        {"tc": "CAPACITY_EFF", "ev": "small plates go thru the slots and hit the sink"},
```

```

    {"tc":"SCENE_ADAPT","ev":"Since we use this in the
    sink"}
  ] // 标签数组; 元素: tc 标签编码 (对应
  return_dim_tag.tag_code) , ev 证据片段 (原文)
}

```

你这版极简 JSON 很清爽 👍

在 Doris 里有两种常见落库方式，建议用**“建结构化表 + 直接 JSON Stream Load 映射字段”**，既省 token 又省 ETL。

方案一（推荐）：建结构化轻量表 + 直接 JSON 导入

1) 建落库表（承接你的字段；tags 用 JSON）

sql

复制代码

```

CREATE TABLE hyy.return_review_fact_min_v1 (
  dt          DATE                                COMMENT '入库日期',
  review_id   VARCHAR(128)                         COMMENT '唯一ID',
  review_source TINYINT                           COMMENT '0=return,1=voc,2=revi...
  review_en    STRING                             COMMENT '原文(英文)',
  review_cn    STRING                             COMMENT '翻译(中文, 可空)',
  sentiment    TINYINT                           COMMENT '-1/0/1',
  tags         JSON                               COMMENT '[{"tc":..., "ev":...}]'
  created_at   DATETIME  DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (review_id) BUCKETS 16
PROPERTIES (
  "dynamic_partition.enable"="true",
  "dynamic_partition.time_unit"="DAY",
  "dynamic_partition.start"="-90",
  "dynamic_partition.end"="7",
  "dynamic_partition.prefix"="p",
  "dynamic_partition.buckets"="16",
  "replication_allocation"="tag.location.default: 3"
)

```

);

-- 可选: 倒排索引便于检索

```
CREATE INDEX idx_rrf_en_inv ON hyy.return_review_fact_min_v1(review_en) USING INV
CREATE INDEX idx_rrf_cn_inv ON hyy.return_review_fact_min_v1(review_cn) USING INV
```

2) 用 Stream Load 直接导入 JSON 文件 (最省事)

假设你把多条这样的 JSON (一行一条或数组) 放在 reviews.json。

一行一条 (NDJSON) 示例

bash

□ 复制代码

```
curl -u user:pass \
-H "label: review_json_${date +%s}" \
-H "format: json" \
-H "read_json_by_line: true" \
-H "columns: dt=curdate(),review_id,review_source,review_en,review_cn,sentiment" \
-T reviews.ndjson \
http://<FE_HOST>:8030/api/hyy/return_review_fact_min_v1/_stream_load
```

JSON 数组文件 (非 NDJSON)

用 jsonpaths 明确字段映射 (按你的键名) :

bash

□ 复制代码

```
curl -u user:pass \
-H "label: review_json_arr_${date +%s}" \
-H "format: json" \
-H "jsonpaths: [\"$.review_id\", \"$.review_source\", \"$.review_en\", \"$.review_"
-H "columns: dt=curdate(),review_id,review_source,review_en,review_cn,sentiment" \
-T reviews.json \
http://<FE_HOST>:8030/api/hyy/return_review_fact_min_v1/_stream_load
```

说明

- “columns 里把 dt 直接设为 curdate()，其他字段与 JSON 键同名即可。”
- “tags 列会原样存为 JSON (无需展开)。”

- “你的极简 JSON 没有 asin、置信度等，可后续升级再加列。”

3) (可选) 把 tags 展开到规范化桥表

若你已经有桥表 hyy.return_review_tag_fact (见前面方案A)，可这样灌：

sql

□ 复制代码

```
INSERT INTO hyy.return_review_tag_fact
(dt, review_id, tag_code, tag_role, evidence_text, confidence, source, model_ver
SELECT
    a.dt, a.review_id,
    json_extract_scalar(t.value, '$.tc') AS tag_code,
    'primary' AS tag_role,
    json_extract_scalar(t.value, '$.ev') AS evidence_
    NULL AS confidence,
    'model' AS source,
    'deepseek-v1' AS model_ver
FROM hyy.return_review_fact_min_v1 a
LATERAL VIEW json_each(a.tags) t
WHERE a.dt = CURDATE();
```

需要主/从时，可让 LLM 多输出一个短键 r (1/0)，再把 CASE WHEN r=1 THEN 'primary' ELSE 'secondary' END 写进上面 SQL。

方案二：原样 JSON 文件入“原始落地表”，再二次 ETL

如果你想完全不建结构化列，也可以先建一张“原始表”，把整条 JSON 当 STRING/JSON 存下来，再通过 SQL 转换到目标表。

1) 原始落地表

sql

□ 复制代码

```
CREATE TABLE hyy.return_review_raw (
    dt DATE,
    js JSON, -- 或 STRING 也行
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
```

```

ENGINE=OLAP
DUPLICATE KEY (dt, created_at)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (created_at) BUCKETS 8;

-- 直接把文件丢进来
-- columns: dt=curdate(), js

```

2) 由 raw → 结构化轻量表

sql

□ 复制代码

```

INSERT INTO hyy.return_review_fact_min_v1
(dt, review_id, review_source, review_en, review_cn, sentiment, tags)
SELECT
    r.dt,
    json_extract_scalar(r.js,'$.review_id'),
    CAST(json_extract_scalar(r.js,'$.review_source') AS TINYINT),
    json_extract_scalar(r.js,'$.review_en'),
    json_extract_scalar(r.js,'$.review_cn'),
    CAST(json_extract_scalar(r.js,'$.sentiment') AS TINYINT),
    json_extract(r.js,'$.tags')
FROM hyy.return_review_raw r
WHERE r.dt = CURDATE();

```

那到底选哪种？

- 要简单直接、可查可用：选**方案一**（结构化轻量表），配 Stream Load 直接映射字段。
- 要最大灵活（原样存档、便于回放）：加一张**raw 表**存整条 JSON，再用 SQL 同步到结构化表（方案二 + 同步SQL）。

最小可运行示例（单条 INSERT）

不走 Stream Load，直接 SQL 插入一条（便于本地试跑）：

sql

□ 复制代码

```

INSERT INTO hyy.return_review_fact_min_v1
(dt, review_id, review_source, review_en, review_cn, sentiment, tags)
VALUES
(
    CURDATE(),
    '702-9563280-0645818_FPL1001M',
    2,
    'This is a revised rating ...',
    '这是更新后的评分 ...',
    -1,
    JSON_PARSE('[
        {"tc": "PARTS_DESIGN", "ev": "doesn\'t hold glasses well; glasses fall over"}, 
        {"tc": "CAPACITY_EFF", "ev": "small plates go thru the slots and hit the sink"}, 
        {"tc": "SCENE_ADAPT", "ev": "Since we use this in the sink"}]
    ')
);

```

需要我给一份 **Stream Load (NDJSON 与数组) 模板文件** 或者 **Airflow 例子** 吗？我可以把上面命令打包好，改下 host/账号就能跑。

↳ 方案二：原样 JSON 文件入“原始落地表”，再二次 ETL

===== 如果你想**完全不建结构化列**，也可以先建一张“原始表”，把整条 JSON 当 `STRING/JSON` 存下来，再通过 SQL 转换到目标表。

后续的ETL处理会在python处理，所以我会选择方案二

很棒，选**方案二**就走“原样 JSON → 原始落地表 → Python ETL 展开”的路径。下面给到**最小可落地**的一套脚本与示例：建表、导入、ETL 展开到目标表（可与前文的 `hyy.return_review_fact_min_v1 / hyy.return_review_tag_fact` 配合使用）。

1) 原始落地表 (只存整条 JSON)

sql

复制代码

```

CREATE TABLE hyy.return_review_raw (
    dt          DATE                      COMMENT '入库日期',
    js          JSON                      COMMENT '整条极简JSON',
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, created_at)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (created_at) BUCKETS 8
PROPERTIES (
    "dynamic_partition.enable" = "true",
    "dynamic_partition.time_unit" = "DAY",
    "dynamic_partition.start" = "-30",
    "dynamic_partition.end" = "7",
    "dynamic_partition.prefix" = "p",
    "dynamic_partition.buckets" = "8",
    "replication_allocation" = "tag.location.default: 3"
);

```

Stream Load 导入原样 JSON

- **NDJSON (一行一条) **最简单:

bash

复制代码

```

curl -u user:pass \
-H "label: raw_ndjson_$(date +%s)" \
-H "format: json" \
-H "read_json_by_line: true" \
-H "columns: dt=curdate(), js" \
-T reviews.ndjson \
http://<FE_HOST>:8030/api/hyy/return_review_raw/_stream_load

```

- JSON 数组文件 (非 NDJSON) :

bash

复制代码

```

curl -u user:pass \
-H "label: raw_array_$(date +%s)" \
-H "format: json" \
-H "jsonpaths: [\"$\"]" \

```

```
-H "columns: dt=curdate(), js" \
-T reviews.json \
http://<FE_HOST>:8030/api/hyy/return_review_raw/_stream_load
```

你的极简 JSON 结构 (示例) :

json

 复制代码

```
{
  "review_id": "702-...FPL1001M",
  "review_source": 2,
  "review_en": "This is a revised rating ...",
  "review_cn": "这是更新后的评分 ...",
  "sentiment": -1,
  "tags": [
    {"tc": "PARTS_DESIGN", "ev": "doesn't hold glasses well; glasses fall over"},
    {"tc": "CAPACITY_EFF", "ev": "small plates go thru the slots and hit the sink"},
    {"tc": "SCENE_ADAPT", "ev": "Since we use this in the sink"}
  ]
}
```

2) 目标表 (结构化) —— 用于被 Python ETL 写入

(若还没建, 建议这两张: 一张“留言主体”, 一张“留言-标签桥表”)

主体表:

sql

 复制代码

```
CREATE TABLE IF NOT EXISTS hyy.return_review_fact_min_v1 (
  dt DATE,
  review_id VARCHAR(128),
  review_source TINYINT,
  review_en STRING,
  review_cn STRING,
  sentiment TINYINT,
  tags JSON,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
```

```

)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (review_id) BUCKETS 16;

CREATE INDEX idx_rrf_en_inv ON hyy.return_review_fact_min_v1(review_en) USING INV

```

桥表：

sql □ 复制代码

```

CREATE TABLE IF NOT EXISTS hyy.return_review_tag_fact (
    dt          DATE,
    review_id   VARCHAR(128),
    tag_code    VARCHAR(64),
    tag_role    VARCHAR(16),
    evidence_text STRING,
    confidence  DECIMAL(5,4),
    source      VARCHAR(16),
    model_version VARCHAR(64),
    created_at   DATETIME DEFAULT CURRENT_TIMESTAMP
)
ENGINE=OLAP
DUPLICATE KEY (dt, review_id, tag_code, tag_role)
PARTITION BY RANGE (dt) ()
DISTRIBUTED BY HASH (review_id) BUCKETS 16;

CREATE INDEX idx_tag_bitmap  ON hyy.return_review_tag_fact(tag_code) USING BITMAP
CREATE INDEX idx_role_bitmap ON hyy.return_review_tag_fact(tag_role) USING BITMAP

```

3) Python ETL (从 raw 读 JSON, 展开并回写)

走 Doris MySQL 协议，依赖 pymysql。逻辑：

1. “从 hyy.return_review_raw 取当天数据；”
2. “解析每条 js；”
3. “写入 return_review_fact_min_v1；”

4. “展开 tags 写入 return_review_tag_fact (默认主标签为 primary , 若后续 LLM 输出带主从可映射) 。”

python

复制代码

```
import json
import pymysql
from datetime import date

DORIS_HOST = "<FE_HOST>"
DORIS_PORT = 9030
USER = "user"
PASS = "pass"
DB = "hyy"

conn = pymysql.connect(host=DORIS_HOST, port=DORIS_PORT, user=USER, password=PASS)
conn.autocommit(True)

cur = conn.cursor()

# 1) 读 raw
cur.execute("""
    SELECT js
    FROM hyy.return_review_raw
    WHERE dt = CURDATE()
""")

rows = cur.fetchall()

# 2) 逐条解析 & 写入
for (js_str,) in rows:
    data = json.loads(js_str)

    rid = data.get("review_id")
    src = data.get("review_source")
    ren = data.get("review_en")
    rcn = data.get("review_cn")
    sen = data.get("sentiment")
    tags = data.get("tags", [])
```

```

# 3) upsert 主体 (这里直接 insert; 如需幂等可改为 delete+insert 或 unique key+pa
cur.execute("""
    INSERT INTO hyy.return_review_fact_min_v1
    (dt, review_id, review_source, review_en, review_cn, sentiment, tags)
    VALUES (CURDATE(), %s, %s, %s, %s, %s, %s)
""", (rid, src, ren, rcn, sen, json.dumps(tags, ensure_ascii=False)))

# 4) 展开标签写桥表
for tag in tags:
    tc = tag.get("tc")
    ev = tag.get("ev")
    # 你的极简JSON没有主从与置信度, 这里默认primary, confidence为空
    cur.execute("""
        INSERT INTO hyy.return_review_tag_fact
        (dt, review_id, tag_code, tag_role, evidence_text, confidence, source, i
        VALUES (CURDATE(), %s, %s, 'primary', %s, NULL, 'model', 'deepseek-v1')
""", (rid, tc, ev))

cur.close()
conn.close()

```

以后如果让 LLM 多吐一个短键 `r` (`1=primary,0=secondary`) 和 `cf` (置信度) , 上面桥表插入时按 CASE 映射即可。

4) 校验/使用示例

看今天入库的主体:

sql

复制代码

```

SELECT review_id, review_source, sentiment, json_length(tags) AS tag_cnt
FROM hyy.return_review_fact_min_v1
WHERE dt = CURDATE();

```

看今天入库的标签明细:

sql

复制代码

```

SELECT review_id, tag_code, tag_role, evidence_text
FROM hyy.return_review_tag_fact
WHERE dt = CURDATE();

```

小结

- **入库：**按你要求，选择“**原样 JSON → raw 表**”，最省心；
- **ETL：**Python 解析原样 JSON，写“主体 + 桥表”，完全掌控字段、幂等与规则；
- **扩展：**以后只要在 JSON 增加极简短键（如 r、cf、asn），无需改 raw 表结构，只改 ETL 即可。



↳ 1) 原始落地表 (只存整条 JSON)

改成这样落库可否？另外，帮忙补充TABLE的COMMENT

```

CREATE TABLE hyy.return_fact_llm (
    review_id      VARCHAR(100) COMMENT '留言ID',
    payload        JSON NULL   COMMENT 'llm output',
    created_at     DATETIME   DEFAULT
                      CURRENT_TIMESTAMP
)
ENGINE=OLAP
UNIQUE KEY(review_id)
COMMENT "
DISTRIBUTED BY HASH(review_id) BUCKETS 6
PROPERTIES (
    "replication_allocation" = "tag.location.default: 3",
    "min_load_replica_num" = "-1",
    "is_being_synced" = "false",
    "storage_medium" = "hdd",
    "storage_format" = "V2",
    "inverted_index_storage_format" = "V1",
    "enable_unique_key_merge_on_write" = "true",
    "light_schema_change" = "true",
    "disable_auto_compaction" = "false",
    "enable_single_replica_compaction" = "false",
)

```

```
"group_commit_interval_ms" = "10000",
"group_commit_data_bytes" = "134217728",
"enable_mow_light_delete" = "false"
);
```

是的，这个落库设计完全可行 👍

你的表 `hyt.return_fact_llm` 可以直接作为 **DeepSeek / 其他 LLM 输出的原始存储层** —— 存储每条留言打标结果的 JSON payload。

我建议给它补全如下 **COMMENT (字段与表级说明)** :

推荐最终版本

sql

 复制代码

```
CREATE TABLE hyt.return_fact_llm (
    review_id    VARCHAR(100) COMMENT '留言唯一ID（如订单号_产品SKU_平台标识，用于幂等写
    payload      JSON NULL      COMMENT 'LLM输出的完整结果JSON，包括原文、翻译、情感、标签
    created_at   DATETIME     DEFAULT CURRENT_TIMESTAMP COMMENT '记录创建时间（数据入层
)
ENGINE=OLAP
UNIQUE KEY(`review_id`)
COMMENT '退货/评论打标原始结果表：存放DeepSeek等LLM返回的JSON结果（原始payload层）'
DISTRIBUTED BY HASH(`review_id`) BUCKETS 6
PROPERTIES (
    "replication_allocation" = "tag.location.default: 3",
    "min_load_replica_num" = "-1",
    "is_being_synced" = "false",
    "storage_medium" = "hdd",
    "storage_format" = "V2",
    "inverted_index_storage_format" = "V1",
    "enable_unique_key_merge_on_write" = "true",
    "light_schema_change" = "true",
    "disable_auto_compaction" = "false",
    "enable_single_replica_compaction" = "false",
    "group_commit_interval_ms" = "10000",
    "group_commit_data_bytes" = "134217728",
```

```

    "enable_mow_light_delete" = "false"
);

```

表设计说明

字段	类型	含义	备注
review_id	VARCHAR(100)	每条留言的唯一 ID	用作唯一键 (UNIQUE KEY) , 支持幂等覆盖更新
payload	JSON	LLM 原始输出内容	存完整 JSON, 对应结构如: {"review_id": "...", "review_source": 2, "review_en": "...", "review_cn": "...", "sentiment": -1, "tags": [{"tc": "...", "ev": "..."}]}
created_at	DATETIME	数据入库时间	Doris 自动填充当前时间

使用方式

1 直接 Stream Load 导入 (推荐)

假设你每天让 Python 调用 LLM 并把结果写成 `llm_results.ndjson` (一行一个 JSON) :

bash

 复制代码

```

curl -u user:pass \
-H "label: llm_payload_$(date +%s)" \
-H "format: json" \
-H "read_json_by_line: true" \
-H "columns: review_id, payload" \
-T llm_results.ndjson \
http://<FE_HOST>:8030/api/hyy/return_fact_llm/_stream_load

```

要求:

- JSON 文件中每行至少有 `"review_id"` ;

- 其余结构（review_source、sentiment、tags等）都嵌在“payload”字段中；
- Doris 自动保存整条 JSON。

2 Python 插入样例

python

复制代码

```

import json, pymysql
conn = pymysql.connect(host='fe_host', port=9030, user='user', password='pass', database='doris')
cur = conn.cursor()

data = {
    "review_id": "702-9563280-0645818_FPL1001M",
    "review_source": 2,
    "review_en": "This is a revised rating ...",
    "review_cn": "这是更新后的评分 ...",
    "sentiment": -1,
    "tags": [
        {"tc": "PARTS_DESIGN", "ev": "doesn't hold glasses well; glasses fall over"},
        {"tc": "CAPACITY_EFF", "ev": "small plates go thru the slots and hit the sink edge when盛不下"},
        {"tc": "SCENE_ADAPT", "ev": "Since we use this in the sink"}
    ]
}

cur.execute("""
    INSERT INTO hyy.return_fact_llm (review_id, payload)
    VALUES (%s, %s)
    ON DUPLICATE KEY UPDATE payload = VALUES(payload)
    """, (data["review_id"], json.dumps(data, ensure_ascii=False)))

conn.commit()
cur.close()
conn.close()

```

3 后续 ETL 拆解

Python 定时任务读取：

sql

 复制代码

```
SELECT review_id, payload
FROM hyy.return_fact_llm
WHERE created_at >= DATE_SUB(NOW(), INTERVAL 1 DAY);
```

然后：

python

 复制代码

```
for rid, payload_json in results:
    p = json.loads(payload_json)
    # 拆出字段后写入结构化表 (review_fact_min_v1, review_tag_fact 等)
```

总结

阶段	表名	功能	优势
原始入库层	hyy.return_fact_llm	存放 LLM 原始输出 JSON	结构极简, IO最小, 支持幂等更新
解析层 (ETL 后)	hyy.return_review_fact_min_v1 / hyy.return_review_tag_fac t	展开 JSON 为结构化字段	用于后续指标统计、打 标分析

建议：保持 hyy.return_fact_llm 仅做“raw ingestion 层”，不要在上面做复杂查询，这样 Stream Load 和 Merge-on-Write 性能最佳。

...

现在我输入下面的信息给你，请你根据终版标签库进行打
标，然后按照return_fact_llm落库，请你给我落库的insert
语句：

review_id: R384TSBX2ZQOS

comment: The design of this is so shockingly bad
that I'm actually angry about it. The mechanism that

makes it sizable causes you to have a "ledge" where dishes can't sit evenly. Even though my sink fits firmly between the sizing the silverware cup doesn't clip on because the bar is in the way. Additionally it doesn't actually sit straight because it's bent out of shape.

下面是基于“终版标签库”打标后的 **payload** (精简字段、仅英文原文、不含可省字段) , 并按你最终选用的 **hyt.return_fact_llm** 表给出的落库 SQL (带幂等覆盖) :

sql

复制代码

```
INSERT INTO hyt.return_fact_llm (review_id, payload)
VALUES (
    'R384TSBX2ZQOS',
    JSON_PARSE('{
        "review_id": "R384TSBX2ZQOS",
        "review_source": 2,
        "review_en": "The design of this is so shockingly bad that I'm actually angry :(",
        "sentiment": -1,
        "tags": [
            {"tc": "STABILITY_TIP", "ev": "sizable mechanism causes a ledge where dishes can't sit evenly"},
            {"tc": "PARTS DESIGN", "ev": "silverware cup doesn't clip on because the bar is in the way"},
            {"tc": "DOA DAMAGED", "ev": "doesn't sit straight; bent out of shape"}
        ]
    }')
)
ON DUPLICATE KEY UPDATE payload = VALUES(payload);
```

说明 (对应打标)

- **STABILITY_TIP**: ...ledge where dishes can't sit evenly → 结构导致放置不平、稳固性不足
- **PARTS DESIGN**: silverware cup doesn't clip on because the bar is in the way → 配件 (刀叉杯/杯架) 与横杆干涉
- **DOA DAMAGED**: doesn't actually sit straight... bent out of shape → 到货即变形/弯曲, 功能性受损

如果之后你希望区分主/从标签或加入置信度，只需在 `tags` 的每个对象中追加短键 `r` (1/0) 、 `cf` (0~1) ， ETL 时再展开即可。



`tags`数组在`tc`后面多加一项，`tag_name_cn`

好的，更新后的落库 SQL (在每个 `tags` 元素中新增 `tag_name_cn`) 如下：

sql

复制代码

```
INSERT INTO hyy.return_fact_llm (review_id, payload)
VALUES (
    'R384TSBX2ZQOS',
    JSON_PARSE('{
        "review_id": "R384TSBX2ZQOS",
        "review_source": 2,
        "review_en": "The design of this is so shockingly bad that I'm actually angry :(",
        "sentiment": -1,
        "tags": [
            {
                "tc": "STABILITY_TIP",
                "tag_name_cn": "稳定性/防倾倒不足",
                "ev": "sizable mechanism causes a ledge where dishes can't sit evenly"
            },
            {
                "tc": "PARTS DESIGN",
                "tag_name_cn": "分区/配件设计不合理（刀叉篮/杯架等）",
                "ev": "silverware cup doesn't clip on because the bar is in the way"
            },
            {
                "tc": "DOA DAMAGED",
                "tag_name_cn": "损坏到货（DOA/受损）",
                "ev": "doesn't sit straight; bent out of shape"
            }
        ]
    }')
```

```
)
ON DUPLICATE KEY UPDATE payload = VALUES(payload);
```

字段对应说明

键名	含义
tc	标签编码 (对应 return_dim_tag.tag_code)
tag_name_cn	中文标签名 (方便直读, 无需 join 维表)
ev	原文证据片段 (触发该标签的句子)

标签匹配依据

- **STABILITY_TIP (稳定性/防倾倒不足)** → “ledge where dishes can't sit evenly” 表示放置不稳定
- **PARTS DESIGN (分区/配件设计不合理)** → “silverware cup doesn't clip on because the bar is in the way” 配件卡位冲突
- **DOA DAMAGED (损坏到货)** → “bent out of shape” 表示到货变形

这一格式与 hyy.return_fact_llm 结构完全兼容, 可直接通过 **Stream Load 或 Python 插入**。



ON DUPLICATE KEY UPDATE payload =
VALUES(payload)
这是什么意思?

这句语法是 **Doris / MySQL 的“幂等写入”机制**, 意思是:

- 👉 当主键 (或唯一键) 已存在时, 不报错, 而是执行**更新 (UPDATE) 操作;
- 👉 当主键不存在时, 执行插入 (INSERT) **操作。

举例说明

以你的表为例：

sql

 复制代码

```
INSERT INTO hyy.return_fact_llm (review_id, payload)
VALUES ('R384TSBX2ZQOS', JSON_PARSE('{"review_id":"R384TSBX2ZQOS","sentiment":-1}')
ON DUPLICATE KEY UPDATE payload = VALUES(payload);
```

逻辑如下：

情况	Doris 行为
这条 review_id = 'R384TSBX2ZQOS' 在表中不存在	执行插入 (INSERT) 新记录
这条 review_id 已存在	<p>不报“重复键错误”，而是执行： <code>UPDATE hyy.return_fact_llm SET payload = VALUES(payload)</code> 即用新的 payload 覆盖旧的那一条</p>

✓ 为什么这么写

1. 保证幂等性

LLM 打标任务可能会多次跑同一条留言。用这句可以确保：

- 首次写入 → 插入
- 之后重跑 → 自动覆盖，不产生重复行。

2. 避免报错

如果不加这句，遇到相同 review_id 会触发“duplicate key error”；

加上后系统可安全更新。

3. 结合 Doris MOW 模式 (Merge-On-Write)

你的表定义里启用了：

sql

 复制代码

```
"enable_unique_key_merge_on_write" = "true"
```

这和 ON DUPLICATE KEY UPDATE 配合，能实现高效的 upsert (插入或更新)。

简化理解

可以把它当作 Doris/MySQL 版的：

python

 复制代码

```
if record_exists(review_id):
    update(payload)
else:
    insert(payload)
```

是否希望我帮你把这段逻辑改成 **Python 批量 upsert** 版本？那样你用 `executemany()` 批量同步 LLM 输出会更高效。

      ...

遗漏了"review_cn": "这是更新后的评分 ..."，// 留言翻译

好的，补上 **中文翻译字段** `review_cn` 后，最终的落库 SQL 如下：

sql

 复制代码

```
INSERT INTO hyy.return_fact_llm (review_id, payload)
VALUES (
    'R384TSBX2ZQOS',
    JSON_PARSE('{
        "review_id": "R384TSBX2ZQOS",
        "review_source": 2,
        "review_en": "The design of this is so shockingly bad that I'm actually angry :(",
        "review_cn": "这个设计糟糕到让我感到生气。可伸缩结构导致出现一个凸起，餐具无法平稳放置。",
        "sentiment": -1,
        "tags": [
            {
                "tc": "STABILITY_TIP",
                "tag_name_cn": "稳定性/防倾倒不足",
                "ev": "sizable mechanism causes a ledge where dishes can't sit evenly"
            }
        ]
    }')
```

```
},
{
  "tc": "PARTS DESIGN",
  "tag_name_cn": "分区/配件设计不合理（刀叉篮/杯架等）",
  "ev": "silverware cup doesn't clip on because the bar is in the way"
},
{
  "tc": "DOA DAMAGED",
  "tag_name_cn": "损坏到货（DOA/受损）",
  "ev": "doesn't sit straight; bent out of shape"
}
```