Week05｜课时 5｜不让 Agent 裸写 SQL：受控指标查询工具 v1 的契约、权限与审计设计

Table of contents

Agent 可以问指标，但不能绕过口径、权限和审计

Week05 的收口不是“让 Agent 更会写 SQL”，而是：

把自由度压到可测试、可授权、可审计的指标查询工具里。

进入实验 回看课时 4 返回 Week05 总览

下载讲义

提供适合离线阅读的 PDF 版和适合批注整理的 Word 版。

PDF 版 · 打印 / 离线阅读 Word 版 · 批注 / 二次整理

这节课解决什么问题

前四讲已经把 transform、tests、docs、lineage 和 metric registry 讲清楚了。

最后一步是回答：

> **Agent 要查指标时，怎样既能回答业务问题，又不突破系统边界？**

答案不是让它自由写 SQL，而是给它一个受控工具：query_support_kpis_v1。

参考学习时间

建议按一节标准课安排：先读懂工具契约，再跑正例和负例。

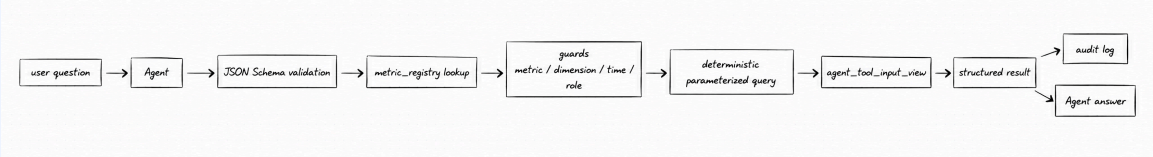
学完这一讲，你应该能做到什么

1. 设计 query_support_kpis_v1 的输入输出 schema。
2. 说明 metric whitelist、dimension whitelist、time window guard、role filter 的作用。
3. 区分工具正例、负例和审计事件。
4. 解释为什么 query tool 应该读 registry，而不是重新发明口径。
5. 把 Week05 自然交给实验和作业页。

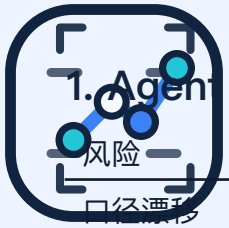
本课产出

- contracts/tools/tools/query_support_kpis_v1.json
- services/tool_api/app/kpi_query.py
- services/tool_api/app/routers/kpis.py
- reports/week05/query_tool_examples.md
- reports/week05/query_tool_audit_notes.md

先看一张总图



工具的价值不是拼 SQL，而是把每一步都变成可检查动作。



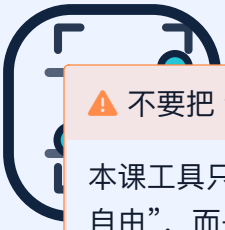
1. Agent 裸写 SQL 的风险

风险	小白版解释	OmniSupport 例子	防线
口径漂移	每次让模型自己想怎么算，结果就会漂	同样问 P1 工单数，一次按 ticket，一次按 event	metric registry 固定指标定义
越权访问	模型可能查到不该看的字段	请求 customer_email、正文、raw comment	role filter + safe view + allowed dimensions
SQL 注入 / 非法 filter	用户输入被拼进查询条件	filter 里塞 raw_sql 或任意表达式	JSON Schema validation + parameterized query
成本失控 / 大窗口查询	一次查太长时间或太多行	查询 365 天、无限 limit	max_window_days + limit
审计缺失	查完以后没人知道谁查了什么	Agent 给了答案，但无法复盘 actor、metric、release	audit payload 记录 actor、metrics、filters、release
不可复现	同样问题下次答不出同样依据	没有 request / release 锚点	actor_id, release_id, data_release_id

> ! 核心判断
>
> 受控工具不是帮 Agent 拼 SQL，而是阻止 Agent 绕过口径、权限、成本和审计边界。

2. query_support_kpis_v1 是什么 / 不是什么

是什么	不是什么
受控指标查询接口	不是 NL2SQL
从 registry 读取指标定义	不是让 Agent 任意选择表和字段
按白名单执行参数化查询	不是 raw SQL 的安全壳
返回结构化结果和 audit	不是完整 Week10 tool / action / HITL 治理系统



⚠ 不要把 tool-safe query 写成 NL2SQL

本课工具只允许查询登记过的指标、维度、过滤器和时间窗口。它的价值不是“让模型更自由”，而是让不该发生的查询被明确拒绝。[1]

3. 输入 schema 怎么收紧

当前项目的工具契约位于 contracts/tools/tools/query_support_kpis_v1.json。输入必须包含：

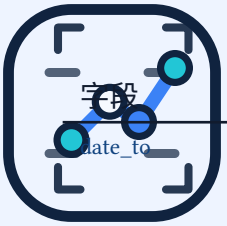
```
{
  "actor_role": "support_ops",
  "actor_id": "ops-demo",
  "metrics": ["p1_ticket_count"],
  "date_from": "2026-04-01",
  "date_to": "2026-04-07",
  "dimensions": ["product_line"],
  "filters": {
    "priority": "p1_critical"
  },
  "limit": 20
}
```

关键约束：

- metrics 只能是 registry 中存在且角色允许的指标；
- dimensions 必须在 allowed_dimensions 中；
- filters 必须在 allowed_filters 中；
- 日期窗口不能超过 registry 的 max_window_days: 31；
- additionalProperties: false，不接受 raw_sql 这类额外字段。

字段	为什么存在	允许值 / 规则	拒绝例	常见拒绝码
actor_role	判断角色能不能查	registry 允许的角色	viewer 查受限指标	ROLE_DENIED
metrics	固定要查哪些指标	registry 中登记且角色允许	raw_sql_revenue	METRIC_DENIED
date_from	查询起点	合法日期，不能晚于 date_to	2026-05-01 到 2026-04-01	INVALID_DATE_RANGE

[1] NL2SQL 让模型生成 SQL；query_support_kpis_v1 让模型只能请求已登记指标，并由 runtime 执行确定性查询。两者风险模型完全不同。



字段	为什么存在	允许值 / 规则	拒绝例	常见拒绝码
date_to	查询终点	合法日期，窗口不超过上限	365 天窗口	WINDOW_TOO_LARGE
dimensions	指标按什么切片	allowed_dimensions 白名单	customer_email	DIMENSION_DENIED
filters	允许哪些条件	allowed_filters 白名单	raw_sql, body_text	FILTER_DENIED
limit	控制返回行数和成本	小于等于工具上限	100000	LIMIT_TOO_LARGE

additionalProperties: false、required、enum 这类严格 schema 思想，能降低输入漂移，但不能替代 registry、role 和 audit。[2]

4. 输出不是自然语言，而是结构化结果

工具输出应该先返回结构化结果，再由 Agent 组织语言。

```
{
 "allowed": true,
 "rows": [
  {
   "metric_date": "2026-04-01",
   "metric_name": "p1_ticket_count",
   "product_line": "Northstar Gateway",
   "metric_value": 12,
   "data_release_id": "week05_local"
  }
 ],
 "denial_code": null,
 "message": null,
 "audit": {
  "tool_name": "query_support_kpis_v1",
  "registry_id": "week05_support_metrics_v1",
  "actor_role": "support_ops",
  "actor_id": "ops-demo",
  "metrics": ["p1_ticket_count"],
  "dimensions": ["product_line"],
  "filters": {"priority": "p1_critical"},
  "date_from": "2026-04-01",
```

[2] Structured Outputs 和 JSON Schema 可以约束模型输出结构；但业务权限、指标白名单和审计仍必须由工具 runtime 执行。

```
  "date_to": "2026-04-07",
  "row_count": 1,
  "release_id": "week05_local"
 }
}
```

失败也必须结构化返回：

```
{
 "allowed": false,
 "rows": [],
 "denial_code": "METRIC_DENIED",
 "message": "metrics are not registered or not role-allowed: raw_sql_revenue",
 "audit": {
  "tool_name": "query_support_kpis_v1",
  "registry_id": "week05_support_metrics_v1",
  "actor_role": "instructor",
  "metrics": ["raw_sql_revenue"]
 }
}
```

audit 字段	它回答的问题
tool_name	这次调用走的是哪个工具
registry_id	查询基于哪版指标契约
actor_role / actor_id	谁以什么身份查
metrics / dimensions / filters	查了什么指标、按什么维度和条件
date_from / date_to	时间窗口是什么
row_count	实际返回多少行
release_id / data_release_id	结果绑定到哪次数据 release
denial_code	如果拒绝，程序可判断的拒绝原因

5. runtime 只做确定性查询

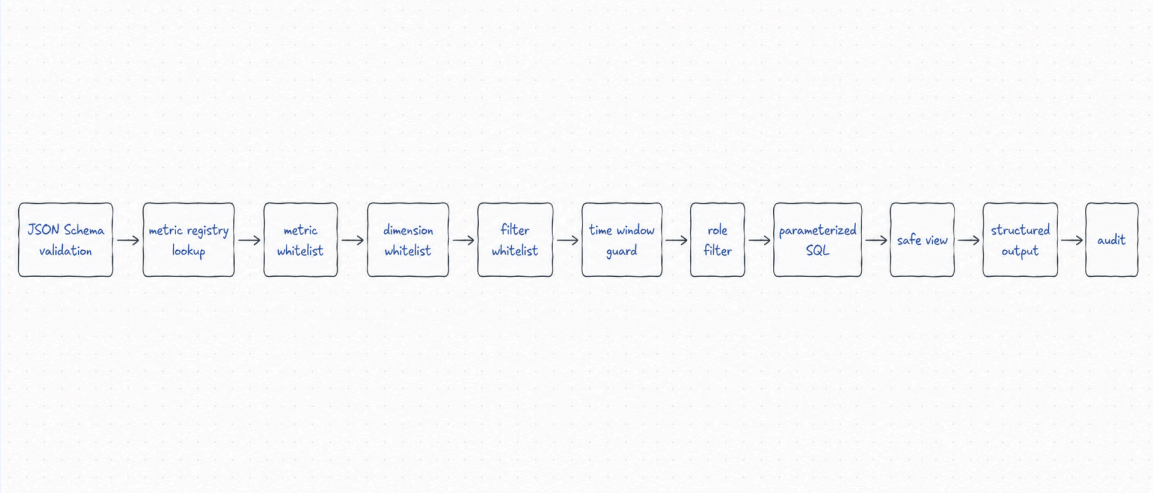
services/tool_api/app/kpi_query.py 的运行路径是固定的：

1. 加载 query_support_kpis_v1.json 的 input schema；
2. 加载 analytics/metric_registry_v1.yml；
3. 校验 actor_role、metric、dimension、filter、date window；
4. 只生成参数化 SQL；

5. 只查询 analytics.agent_tool_input_view；
6. 返回 rows 和 audit；
7. 任何失败都用 denial_code 表达。

它不会：

- 接收 raw SQL；
- 读取 raw source table；
- 让 Agent 指定任意列；
- 自动扩展到未登记指标；
- 静默吞掉拒绝原因。

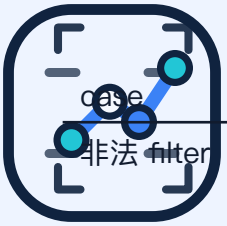


参数化查询只能防一部分注入风险，不能替代权限、口径和时间窗口白名单。[3]

6. 正例 / 负例表

case	输入	期望
合法指标 + 合法维度	p1_ticket_count by product_line	返回结果，allowed=true
合法指标 + 多维切片	ticket_count by product_line, priority	返回安全视图字段
未知指标	secret_revenue 或 raw_sql_revenue	拒绝，METRIC_DENIED
非法维度	customer_email	拒绝，DIMENSION_DENIED
过大时间窗口	365 天	拒绝，WINDOW_TOO_LARGE
角色越权	viewer 查询 registry 指标	拒绝，ROLE_DENIED

[3] 参数化 SQL 能降低注入风险，但不能证明用户有权限查、指标口径正确、时间窗口合理或结果可审计。



case	输入	期望
非法 filter	raw_sql	拒绝，FILTER_DENIED 或 schema validation failed
limit 过大	limit: 100000	拒绝，LIMIT_TOO_LARGE 或 schema validation failed
日期反向	date_from 晚于 date_to	拒绝，INVALID_DATE_RANGE
mart 有但 registry 未开放	avg_first_response_minutes	拒绝，METRIC_DENIED 或暂不开放

当前项目 runbook 已提供 CLI 正负例：

```
docker compose --profile tools --env-file infra/env/.env.local \
  -f infra/docker-compose.yml run --rm devbox \
  bash -lc 'PYTHONPATH=services/tool_api python -m app.kpi_query --example valid'

docker compose --profile tools --env-file infra/env/.env.local \
  -f infra/docker-compose.yml run --rm devbox \
  bash -lc 'PYTHONPATH=services/tool_api python -m app.kpi_query --example bad_metric || true'

docker compose --profile tools --env-file infra/env/.env.local \
  -f infra/docker-compose.yml run --rm devbox \
  bash -lc 'PYTHONPATH=services/tool_api python -m app.kpi_query --example bad_role || true'
```

项目证据显示：正例 allowed=true，未知指标返回 METRIC_DENIED，角色越权返回 ROLE_DENIED。

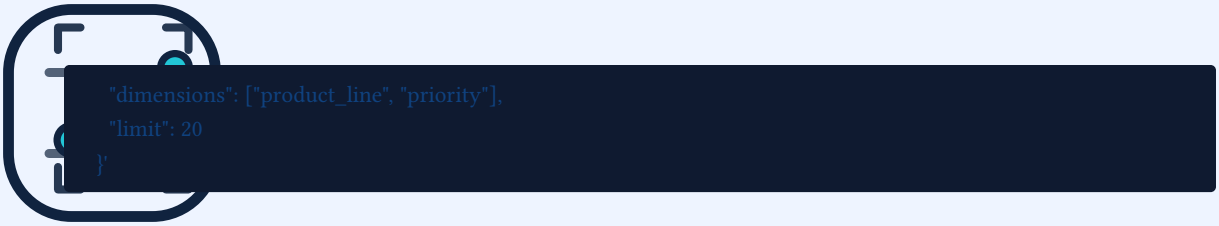
7. Tool API 端点怎么验证

如果要验证服务端 endpoint：

```
docker compose --env-file infra/env/.env.local \
  -f infra/docker-compose.yml up -d --build tool_api

curl -s http://localhost:8001/health

curl -s -X POST http://localhost:8001/api/v1/tools/query_support_kpis \
  -H 'Content-Type: application/json' \
  -H 'X-Actor-ID: instructor-local' \
  -d '{
    "actor_role": "instructor",
    "metrics": ["ticket_count"],
    "date_from": "2026-04-01",
    "date_to": "2026-04-30",
```



8. Week10 会继续什么，本周不抢什么

Week10 会继续扩展 tool governance、HITL、更多 action tools 和审计策略。[4]

Week05 只做一件事：让“查客服运营指标”这个动作先变成受控、可测试、可审计的查询工具 v1。

自检清单

- ☐ 我能画出受控指标查询路径。
- ☐ 我知道哪些输入必须被 schema 限制。
- ☐ 我能列出至少 5 个负例。
- ☐ 我知道 audit log 最少要记录什么。
- ☐ 我能说明工具为什么必须复用 metric registry。

课后最小行动

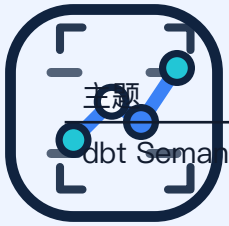
在你的 Week05 笔记里写出 3 条查询例子：

- 一个应当通过的查询；
- 一个 metric 不存在的拒绝例子；
- 一个非法维度或过大时间窗口的拒绝例子。

延伸阅读

主题	推荐资料	为什么读
OpenAI Function Calling	OpenAI Docs: Function Calling	理解 Agent 如何通过工具契约调用能力
OpenAI Structured Outputs	OpenAI Docs: Structured Outputs	理解严格结构化输出和 JSON Schema 的关系
JSON Schema	JSON Schema: Understanding JSON Schema	理解 required、enum、additionalProperties 等约束

[4] Week10 会继续展开更多 tool/action/HITL 治理。Week05 只把“安全查询客服运营指标”这一个动作做成 v1 工程接口。



主题	推荐资料	为什么读
dbt Semantic Layer	dbt Docs: Semantic Layer	回看指标来源边界为什么应由语义层或 registry 提供
OWASP SQL Injection Prevention	OWASP Cheat Sheet: SQL Injection Prevention	理解参数化查询的安全价值和边界

Footnotes