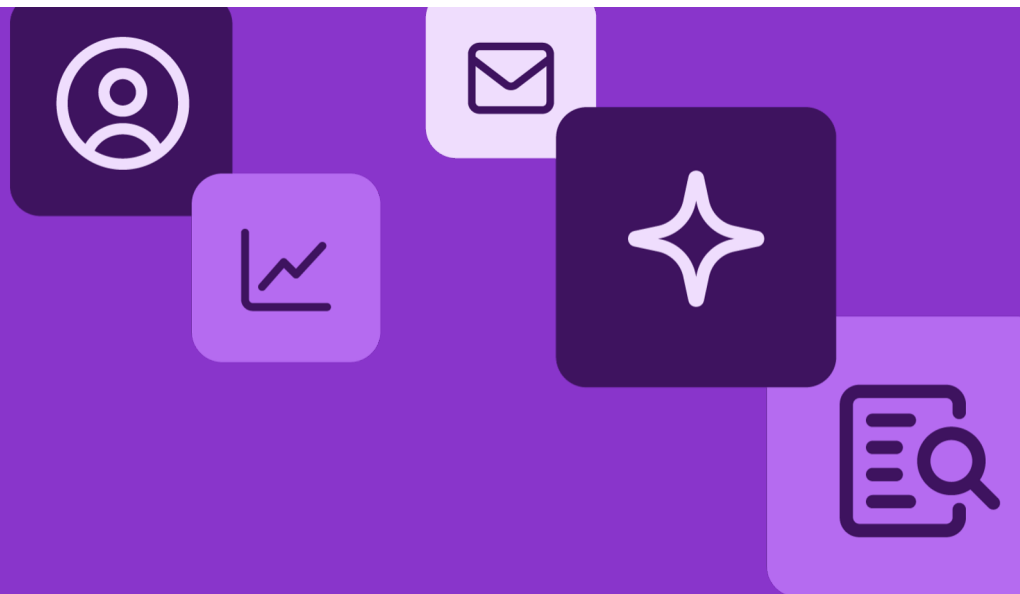




现代化的订单管理

免责声明：
本文内容用于解释eBay平台的各项操作流程或平台政策、规则。本文中的任何（包括隐含的）观点、建议、结论等仅供参考，不构成对您的最终操作的建议。
相关政策、规则的正式版本以各平台独立公布的为准。eBay不就本文中的内容等对您的最终操作等做出任何保证或者承诺，您据此做出的任何决策与eBay公司及编制人员无关。

01 订单流程

02 订单和售后调用流程

03 订单连接点+售后

04 售后流程

05 不同类型的合约取消

06 退货流程

07 售后询问流程

**08 Trading API vs GraphQL:
订单 API**

**09 Trading API vs GraphQL:
售后 API**

10 GraphQL查询示例

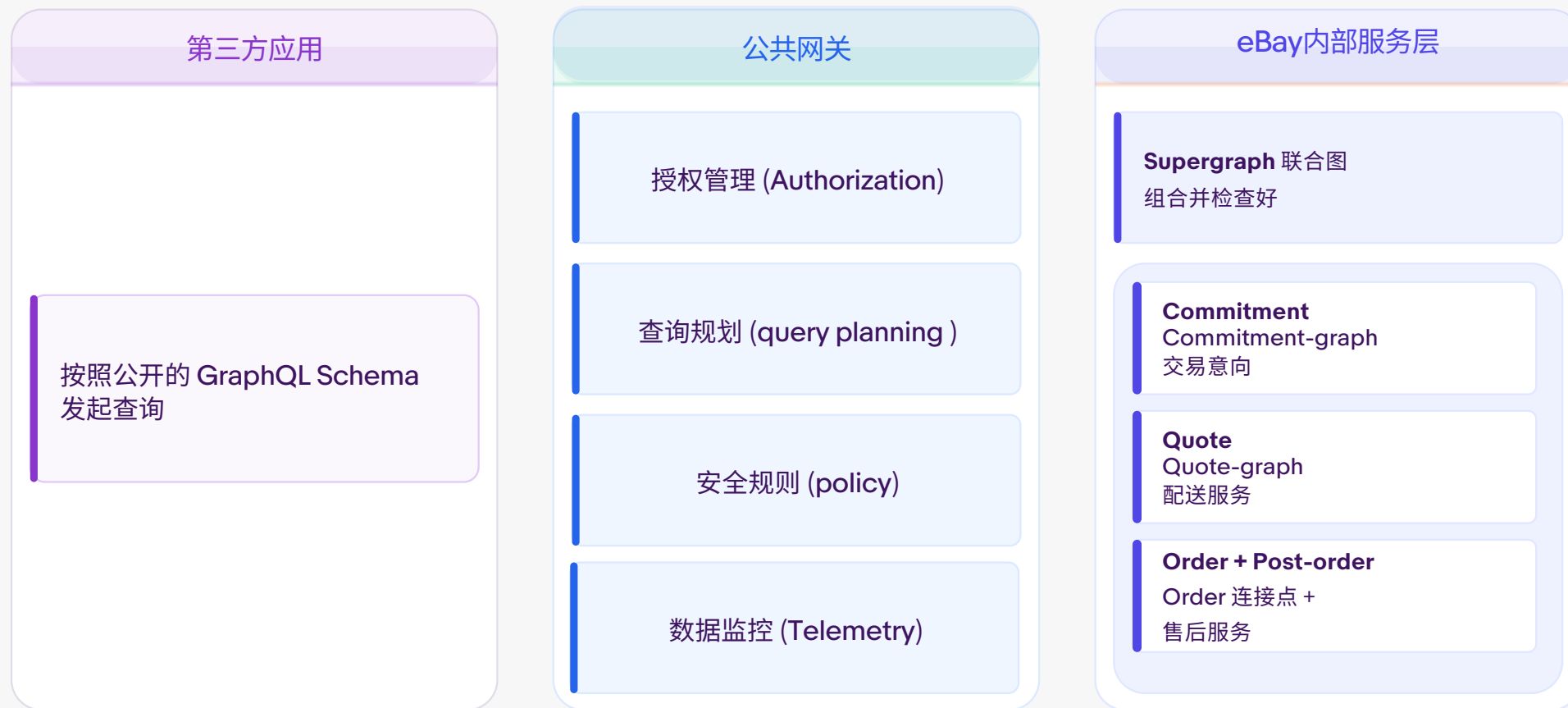
目录

1. 订单流程

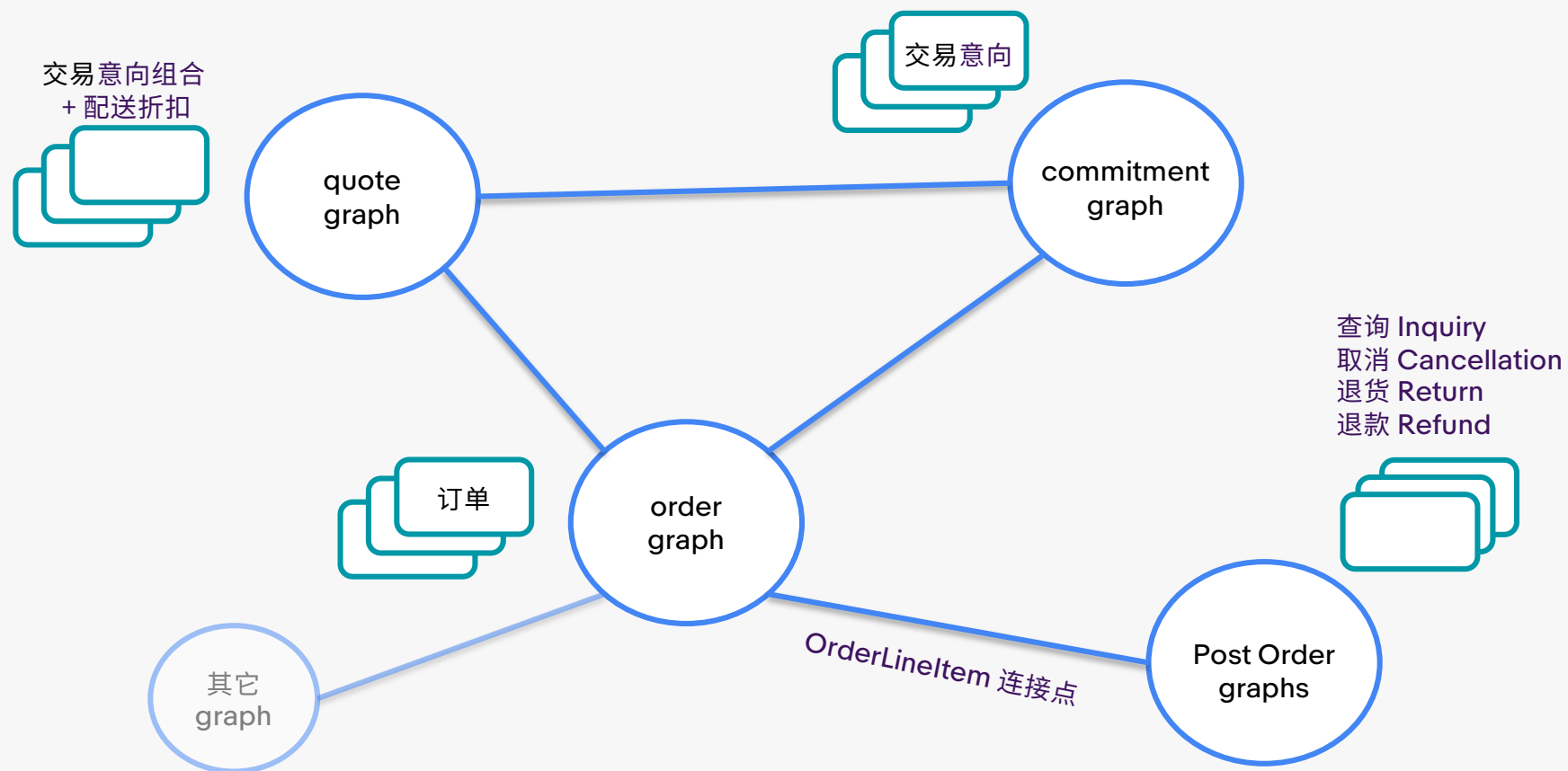
承诺 → 报价 → 订单 → 售后

Commitment	PurchaseQuote	Order	Post-order
承诺 - 买家的购买意向 - 卖家希望出售的库存	购买报价 - 承诺 - 相同的买家和卖家 - 配送方案	订单 - 结账时创建订单 连接点 - Stable Order + - line IDs	售后 - 查询 - 取消 - 退货 - 退款

2. 订单和售后调用流程



3. 订单连接点+ 售后



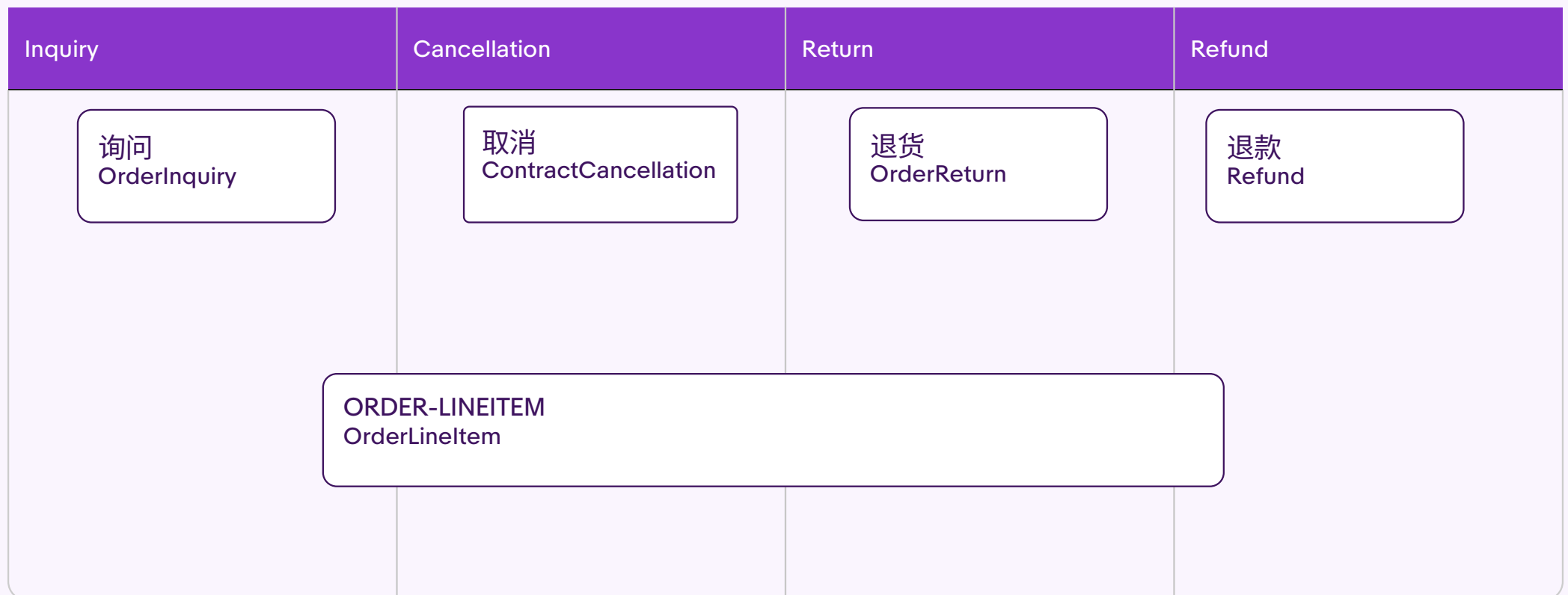
- 客户端通过一个统一的 Graph 进行访问，同时各领域保持各自的负责权

Entity links
实体关联

OrderLineItem
连接查询、取消、退货和退款等售服务

4. 售后流程

Order 和 OrderLineItem 始终作为连接点



5. 不同类型的合约取消

1 接口

- cancellationId
- reason
- status



2 合约取消

ItemCommitmentCancellation
将取消操作与 Item Commitment 的来源关联起来。

PurchaseQuoteCancellation
将取消操作与 Purchase Quote 的来源关联起来。

OrderCancellation
将取消操作与订单中的 OrderLineItem 关联起来。



3 共享状态

所有这些取消类型都共享：

- 买家/卖家操作
- 状态
- 退款

6. 退货流程：卖家的响应由当前状态决定

OrderReturn 是面向卖家的核心聚合对象；GraphQL 根据当前状态提供有效的响应操作；客户端执行相应操作，并跟随之后产生的对应类型流程。

卖家客户端如何使用 GraphQL

1 查找待处理事项

sellerOrderReturns

默认队列会优先显示等待卖家处理的退货。

2 查看退货

OrderReturn

查看当前状态、商品明细、退款金额预估、当前正在进行的子流程。

3 显示可执行操作

sellerAvailableActions

根据当前状态和卖家角色，显示：主要可执行操作截止时间可选的次要操作。

4 执行操作并刷新

mutation → OrderReturn

执行系统提供的操作，然后根据更新后的状态重新查询并显示。

根据当前状态，卖家可以采取的处理路径

TYPED CHILD PROCESSES ● ReturnFulfillment ● ReturnNegotiation ● Refund ● PostPurchaseCase

A 批准 / 启动退货履约

当需要卖家做出决定时，批准退货。进入 OPEN 状态后，如果需要 RMA 编号，或者 eBay 无法提供退货物流标签，则由卖家提供相关信息，以启动 ReturnFulfillment（退货履约）流程。

卖家操作

approveReturn
provideRmaNumber
attachReturnShipmentLabel

追踪退货流程

returnFulfillment.status
details → buyer initiation → in progress

ReturnFulfillment 负责实际的退货物流流程

N 协商

提供部分退款，让买家保商品，而不是把商品退回来。

卖家操作

proposeKeepItemWithPartialRefund

追踪退货后续状态

returnNegotiation
.latestReturnOffer
.proposedTerm[] / .status

ReturnNegotiation 负责退款方案的条款和状态

R 退款

根据退货政策立即退款，或者等退回的商品收到后再退款。

卖家操作

processReturnRefund

追踪资金流向

refunds[] → refundStatus
sellerRefundBreakdown
fundings[] / refundRecoveries[]

Refund owns money movement and execution details

C 转交给 eBay

当需要 eBay 介入并处理退货时，提交转交请求。

卖家操作

submitReturnReferral

追踪个案

cases[] → PostPurchaseCase
status · latestDecision · decisionHistory[]

PostPurchaseCase 负责转交后的决定和审核

拒绝 / 发送消息

根据当前状态和卖家角色，在系统提供这些选项时，可以：
declineReturn | sendReturnMessage

客户端规则

显示 sellerAvailableActions → 执行系统提供的 mutation → 然后重新查询 OrderReturn。

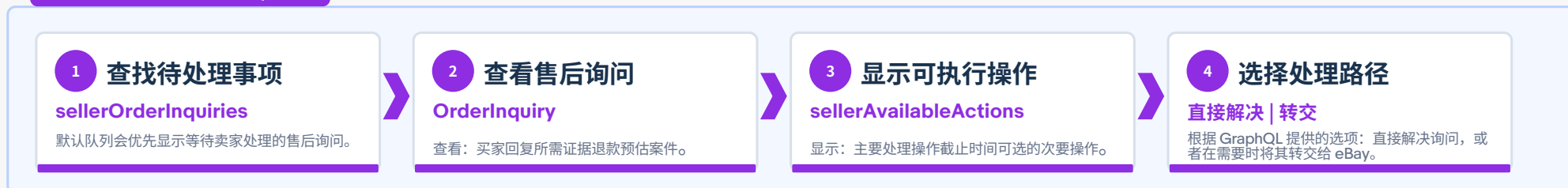
买家响应

买家对卖家的退款方案做出回应，或者使用卖家提供的物流标签启动 ReturnFulfillment。

7. 售后询问流程：直接解决或转交给 eBay

售后询问决定卖家应采取的处理方式。直接提供的证据和退款信息会保留在该售后询问中；转交后会创建一个 `PostPurchaseCase`，并记录决定历史。

卖家客户端如何使用 GraphQL



直接解决

卖家直接解决询问，无需创建 Case

该售后询问本身会作为整个处理流程的记录，用于保存证据、退款信息、状态以及关闭信息。

- ✓ 提供物流信息或证据
`provideShipmentInfoForItemNotReceivedInquiry` → `providedInformation[]`
- ✓ 符合条件时发起全额退款
`processOrderInquiryRefund` → `refunds[]`
- ✓ 发生延迟时进行沟通
`sendOrderInquiryMessage`

转交与决策的生命周期

转交后会创建一个具有明确类型的 `PostPurchaseCase`

卖家从 GraphQL 返回的可选原因中选择一个；当需要 eBay 介入解决该售后询问时，提交转交请求。

1. 选择转交原因
`referralReasons[]`
只显示当前符合条件的转交原因
2. 提交
`submitOrderInquiryReferral`
返回更新后的售后询问
3. Follow case
`cases[]`
只有转交后才会出现。

`PostPurchaseCase` 状态: OPEN / ON_HOLD / CLOSED
`latestDecision + decisionHistory[]` `inFavorOf • outcomes[] • optional reviewRequest`

共享模型

`Return` 和售后询问只有在转交后才会显示 `cases[]`。父级对象仍然是卖家的入口；`PostPurchaseCase` 负责管理审核状态以及已经执行的决定。

8. Trading API vs GraphQL: 订单API

类型化的生命周期实体，替代传统以交易记录为核心的查询模型

Entity	Trading API	GraphQL
Order	GetOrders 结账完成后的订单中心化数据获取	<pre>orders { sellerOrders(input: \$orders) { orders { orderId orderState { __typename } } } }</pre>
PurchaseQuote	没有独立的 Purchase Quote（购买报价）合集通过 Transaction（交易）或 Order（订单）来查看购买流程后续阶段的信息	<pre>purchaseQuotes { sellerPurchaseQuotes(input: \$quotes) { purchaseQuotes { purchaseQuoteId state { __typename } } } }</pre>
ItemCommitment	GetItemTransactions 用于获取购买产生的交易记录 它不会将购买前的意向作为一个独立的合集来建模	<pre>itemCommitments { sellerItemCommitments(input: \$commitments) { itemCommitments { itemCommitmentId state { __typename } } } }</pre>

查询输入支持首页过滤和排序，也支持基于游标的分页，例如：OrdersInput、PurchaseQuotesInput、ItemCommitmentsInput。

9. Trading API vs GraphQL: 售后API

过去较为笼统的操作，会被转变为具有明确类型的 FGQL 查询 (queries) 和变更操作 (mutations)

Flow	Trading API	GraphQL
Inquiry retrieval	Search Inquiries Get Inquiry	sellerOrderInquiries orderInquiry
Inquiry resolution	Provide shipment info Issue refund Send message	provideShipmentInfoForItemNotReceivedInquiry processOrderInquiryRefund sendOrderInquiryMessage
Inquiry referral	Escalate Inquiry Appeal Inquiry	submitOrderInquiryReferral Referral reasons depend on current state
Return retrieval	Search Returns Get by Return ID	sellerOrderReturns orderReturn
Return decision	Process Return with a decision enum Issue Refund	approveReturn or declineReturn proposeKeepItemWithPartialRefund processReturnRefund
Return fulfillment	Add or mark shipping label Mark received Escalate Return	attachReturnShipmentLabel recordReturnLabelSentOffline recordReturnReceived or submitReturnReferral

GraphQL查询示例

10.1 ItemCommitment 查询示例

基于 Schema 的查询，并返回组合式的 ID 和值。

查询

```
query CommitmentExample {
  itemCommitmentsByIds(input: {
    itemCommitmentIds: ["99-000-0001001"]
  }) {
    __typename
    ... on ItemCommitmentsByIdsSuccess {
      itemCommitments {
        itemCommitmentId
        createdAt
        state {
          __typename
          ... on Labeled { label }
        }
      }
    }
  }
}
```

组合式响应

```
{
  "data": {
    "itemCommitmentsByIds": {
      "__typename":
        "ItemCommitmentsByIdsSuccess",
      "itemCommitments": [{
        "itemCommitmentId": "99-000-0001001",
        "createdAt": "2026-08-12T18:30:00Z",
        "state": {
          "__typename": "ItemCommitmentActive",
          "label": "Active"
        }
      }]
    }
  }
}
```

10.2 PurchaseQuote 查询示例

基于 Schema 的查询，并返回组合式的 ID 和值。

查询

```
query PurchaseQuoteExample {
  purchaseQuotesByIds(input: {
    purchaseQuoteIds: ["99-00002-00002"]
  }) {
    __typename
    ... on PurchaseQuotesByIdsSuccess {
      purchaseQuotes {
        purchaseQuoteId
        createdAt
        lineItems { purchaseQuoteLineItemId }
        state {
          __typename
          ... on Labeled { label }
        }
      }
    }
  }
}
```

组合式响应

```
{
  "data": {
    "purchaseQuotesByIds": {
      "__typename": "PurchaseQuotesByIdsSuccess",
      "purchaseQuotes": [{
        "purchaseQuoteId": "99-00002-00002",
        "createdAt": "2026-08-12T18:32:00Z",
        "lineItems": [{
          "purchaseQuoteLineItemId":
            "99-00002-00452"
        }],
      }],
      "state": {
        "__typename": "PurchaseQuoteActive",
        "label": "Active"
      }
    }
  }
}
```

10.3 Order 查询示例

基于 Schema 的查询，并返回组合式的 ID 和值。

查询

```
query OrderExample {
  ordersByIds(input: {
    orderIds: ["98-00002-00002"]
  }) {
    __typename
    ... on OrdersByIdsSuccess {
      orders {
        orderId
        confirmedAt
        orderLineItems { orderId }
        orderState {
          __typename
          ... on Labeled { label }
        }
      }
    }
  }
}
```

组合式响应

```
{
  "data": {
    "ordersByIds": {
      "__typename": "OrdersByIdsSuccess",
      "orders": [{
        "orderId": "98-00002-00002",
        "confirmedAt": "2026-08-12T18:36:00Z",
        "orderLineItems": [{
          "orderLineItemId": "980000200087"
        }],
        "orderState": {
          "__typename": "OrderConfirmed",
          "label": "Confirmed"
        }
      }]
    }
  }
}
```

10.4 Inquiry 查询示例

基于 Schema 的查询，并返回组合式的 ID 和值。

查询

```
query InquiryExample {  
  orderInquiry(input: {  
    inquiryId: "9000004001"  
  }) {  
    __typename  
    ... on OrderInquirySuccess {  
      orderInquiry {  
        inquiryId  
        createdAt  
        initiator  
        responder  
        status {  
          __typename  
          code  
          displayText  
        }  
      }  
    }  
  }  
}
```

组合式响应

```
{  
  "data": {  
    "orderInquiry": {  
      "__typename": "OrderInquirySuccess",  
      "orderInquiry": {  
        "inquiryId": "9000004001",  
        "createdAt": "2026-08-20T16:05:00Z",  
        "initiator": "BUYER",  
        "responder": "SELLER",  
        "status": {  
          "__typename": "OrderInquiryOpen",  
          "code": "OPEN",  
          "displayText": "Open"  
        }  
      }  
    }  
  }  
}
```

10.5 Cancellation 查询示例

查询

```
query CancellationExample {
  contractCancellation(input: {
    cancellationId: "9000005001"
  }) {
    __typename
    ... on ContractCancellationSuccess {
      contractCancellation {
        __typename
        cancellationId
        requestedAt
        requester
        reason { __typename code displayText }
        status { __typename code displayText }
      }
    }
  }
}
```

基于 Schema 的查询，并返回组合式的 ID 和值。

组合式响应

```
{
  "data": {
    "contractCancellation": {
      "__typename": "ContractCancellationSuccess",
      "contractCancellation": {
        "__typename": "OrderCancellation",
        "cancellationId": "9000005001",
        "requestedAt": "2026-08-21T09:10:00Z",
        "requester": "BUYER",
        "reason": {
          "__typename": "BuyerRequestedCancellation",
          "code": "BUYER_REQUESTED",
          "displayText": "Buyer requested"
        },
        "status": {
          "__typename": "ContractCancellationPending",
          "code": "PENDING",
          "displayText": "Pending"
        }
      }
    }
  }
}
```

10.6 Return 查询示例

查询

```
query ReturnExample {
  orderReturn(input: {
    returnId: "9000006001"
  }) {
    __typename
    ... on OrderReturn {
      returnId
      requestedAt
      requester
      status { __typename code displayText }
      returnLineItems {
        returnLineItemId
        requestedQuantity
        orderLineItem { orderLineItemId }
      }
    }
  }
}
```

基于 Schema 的查询，并返回组合式的 ID 和值。

组合式响应

```
{
  "data": {
    "orderReturn": {
      "__typename": "OrderReturn",
      "returnId": "9000006001",
      "requestedAt": "2026-08-25T14:12:00Z",
      "requester": "BUYER",
      "status": {
        "__typename": "ReturnOpen",
        "code": "OPEN",
        "displayText": "Open"
      },
      "returnLineItems": [{
        "returnLineItemId": "9000006005",
        "requestedQuantity": 1,
        "orderLineItem": {
          "orderLineItemId": "980000200002"
        }
      }]
    }
  }
}
```

10.7 Refund 查询示例

基于 Schema 的查询，并返回组合式的 ID 和值。

查询

```
query RefundExample {
  orderReturn(input: {
    returnId: "9000006002"
  }) {
    __typename
    ... on OrderReturn {
      returnId
      refunds {
        refundId
        refundStatus {
          __typename
          code
          displayText
        }
      }
    }
  }
}
```

组合式响应

```
{
  "data": {
    "orderReturn": {
      "__typename": "OrderReturn",
      "returnId": "9000006002",
      "refunds": [{
        "refundId": "9000007001",
        "refundStatus": {
          "__typename": "RefundCompleted",
          "code": "COMPLETED",
          "displayText": "Completed"
        }
      }]
    }
  }
}
```



谢谢

