**DQL Tutorial for Event Analysis & Operational Monitoring**

**1️.Basic Event Queries**

📍 Fetch All Events

fetch events

📍 Fetch Events of Type "DAVIS\_EVENT"

fetch events

| filter event.kind == "DAVIS\_EVENT"

**📍 Fetch Events from the Last 10 Minutes**

fetch events, from:now()-10m

**Sorting & Filtering Events**

**📍 Fetch Infrastructure-Related Events & Sort by Host**

fetch events, from:now()-10m

| filter dt.davis.impact\_level == "Infrastructure"

| sort dt.entity.host.name

 Filters **only infrastructure-related events**.

 Sorts results by **host name**.

📍 Fetch Only Recovered Events

fetch events, from:now()-10m

| filter event.status\_transition == "RECOVERED"

| fields dt.entity.host.name, event.name, event.status\_transition, event.status

 Filters **only recovered events**.

 Displays **host name, event name, transition, and status**.

**Aggregation & Summarization**

**📍 Count Events by Type**

fetch events

| summarize count(), by:event\_type

* Groups **events by type** and counts occurrences.

📍 Count Distinct Problems (Top 20)

fetch events

| filter event.kind == "DAVIS\_PROBLEM"

| summarize {problemCount = countDistinct(event.id), by:{event.name}}

| sort problemCount desc

| limit 20

 Filters **problems only**.

 Counts **distinct problem IDs**.

 Sorts in **descending order**.

 Limits **top 20 problems**.

**📍 Summarize Closed Problems by Event Name**

fetch dt.davis.problems

| filter event.status == "CLOSED"

| summarize by:{event.name}, count = count()

 Filters **only closed problems**.

 Groups by **event name** and counts occurrences.

📍 Summarize Open Problems by Event Name

fetch dt.davis.problems

| filter event.status == "ACTIVE"

| summarize by:{event.name}, count = count()

 Filters **only active problems**.

 Groups by **event name** and counts occurrences.

**Time-Series Analysis**

**📍 Show Event Count Over Time (1-Minute Interval)**

fetch events

| makeTimeseries count(), time:{timestamp}, interval:1m

Aggregates **event count over time** in **1-minute intervals**.

**Advanced Event Filtering**

**📍 Fetch Only Davis Problems**

fetch events

| filter event.kind == "DAVIS\_PROBLEM"

📍 Show Most Impacted Entities (Top 10)

fetch events

| filter event.kind == "DAVIS\_PROBLEM"

| expand affected\_entity\_ids

| summarize by:{affected\_entity\_ids}, count = countDistinct(display\_id)

| sort count desc

| limit 10

 Filters **only Davis problems**.

 Expands **affected entities**.

 Counts **distinct problem IDs per entity**.

 Sorts in **descending order** and **limits top 10**.

**Operational Monitoring Use Cases**

**📍 Find Open Problems Affecting Cloud Applications**

fetch events

| sort timestamp desc

| filter dt.source\_entity.type == "cloud\_application"

| filter event.status == "ACTIVE"

 Filters **only cloud application issues**.

 Finds **active problems**.

📍 Show Events with Their Impact Level

fetch dt.davis.events

| fields timestamp, dt.davis.impact\_level, dt.source\_entity, entity\_tags, event.name, dt.entity.host

| summarize count(), by:{event.name}

 Displays **event impact level**.

 Groups by **event name** and **counts occurrences**.

📍 Chart Problem Counts by Impact Level Over Time

fetch events

| filter event.kind == "DAVIS\_PROBLEM"

| makeTimeseries by:{dt.davis.impact\_level}, interval:1h, count = count()

 Filters **only Davis problems**.

 Groups by **impact level**.

 Creates a **time-series view** in **1-hour intervals**.

**Application-Specific Event Analysis**

**📍 Fetch Travel Booking Payment Events**

fetch events

| filter event.type == "travel.funnel.booking-payment"

📍 Fetch Product Details from Booking Events

fetch events

| filter event.type == "travel.funnel.booking-payment"

| fields product

📍 Find Booking Events with Platinum Loyalty Customers & Children Travelers

fetch events

| filter event.type == "travel.funnel.booking-payment"

| filter loyaltyStatus == "Platinum" and childrenTravelers > 0

📍 Summarize Booking Revenue by Travel Agency

fetch events

| filter event.type == "travel.funnel.booking-payment"

| summarize sum = sum(amount), by:travelAgency