DerivInsight - Query Reference Guide

Natural Language to SQL Mapping

This document contains sample queries organized by domain. Each query shows:

- **NL Query**: What the user might ask
- **SQL**: The corresponding database query
- **Chart**: Recommended visualization

🔵 COMPLIANCE QUERIES

1. KYC Status Overview

NL: "Show me KYC status breakdown"

```sql
SELECT
    kyc_status,
    COUNT(*) as count,
    ROUND(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM users), 1) as percentage
FROM users
GROUP BY kyc_status
ORDER BY count DESC;
```

Chart: Pie Chart

---

2. Pending KYC Verifications

NL: "Show all pending KYC verifications"

```sql
```

```
SELECT
    user_id,
    full_name,
    email,
    country,
    created_at,
    JULIANDAY('now') - JULIANDAY(created_at) as days_pending
FROM users
WHERE kyc_status = 'PENDING'
ORDER BY created_at ASC;
```

Chart: Table

---

3. High-Risk Users Needing Attention

NL: "List high-risk users with incomplete documentation"

```sql
SELECT
    u.user_id,
    u.full_name,
    u.country,
    u.risk_level,
    u.risk_score,
    u.kyc_status,
    u.is_pep,
    COUNT(a.alert_id) as open_alerts
FROM users u
LEFT JOIN alerts a ON u.user_id = a.user_id AND a.status = 'OPEN'
WHERE u.risk_level = 'HIGH'
GROUP BY u.user_id
ORDER BY u.risk_score DESC, open_alerts DESC;
```

Chart: Table

---

4. Flagged Transactions Over Threshold

NL: "Show flagged transactions over $50,000 in last 30 days"

```sql
```

```
SELECT
    t.txn_id,
    t.user_id,
    u.full_name,
    t.amount_usd,
    t.txn_type,
    t.flag_reason,
    t.created_at
FROM transactions t
JOIN users u ON t.user_id = u.user_id
WHERE t.status = 'FLAGGED'
  AND t.amount_usd > 50000
  AND t.created_at >= datetime('now', '-30 days')
ORDER BY t.amount_usd DESC;
```

Chart: Table

---

5. Transaction Volume by KYC Status

NL: "Transaction volume by KYC status"

```sql
SELECT
    u.kyc_status,
    COUNT(t.txn_id) as txn_count,
    ROUND(SUM(t.amount_usd), 2) as total_volume,
    ROUND(AVG(t.amount_usd), 2) as avg_amount
FROM users u
JOIN transactions t ON u.user_id = t.user_id
GROUP BY u.kyc_status
ORDER BY total_volume DESC;
```

Chart: Bar Chart

---

6. Users from High-Risk Jurisdictions

NL: "Users from FATF high-risk countries"

```sql
```

```
SELECT
    u.user_id,
    u.full_name,
    u.country,
    u.risk_level,
    u.kyc_status,
    COUNT(t.txn_id) as txn_count,
    COALESCE(SUM(t.amount_usd), 0) as total_volume
FROM users u
LEFT JOIN transactions t ON u.user_id = t.user_id
WHERE u.country IN ('AF', 'IR', 'KP', 'SY', 'YE', 'MM', 'PK')
GROUP BY u.user_id
ORDER BY total_volume DESC;
```

Chart: Table

---

7. Expired KYC Documents

NL: "Users with expired KYC documents"

```sql
SELECT
    user_id,
    full_name,
    email,
    country,
    kyc_expiry_date,
    JULIANDAY('now') - JULIANDAY(kyc_expiry_date) as days_expired
FROM users
WHERE kyc_status = 'EXPIRED'
   OR (kyc_expiry_date IS NOT NULL AND kyc_expiry_date < date('now'))
ORDER BY kyc_expiry_date ASC;
```

Chart: Table

---

8. PEP (Politically Exposed Persons) Activity

NL: "Show all PEP accounts and their activity"

```sql
```

```
SELECT
    u.user_id,
    u.full_name,
    u.country,
    u.risk_score,
    COUNT(t.txn_id) as txn_count,
    COALESCE(SUM(t.amount_usd), 0) as total_volume,
    MAX(t.created_at) as last_activity
FROM users u
LEFT JOIN transactions t ON u.user_id = t.user_id
WHERE u.is_pep = 1
GROUP BY u.user_id
ORDER BY total_volume DESC;
```

Chart: Table

---

9. Structuring Detection (Multiple transactions just below threshold)

NL: "Detect potential structuring - transactions just below $10,000"

```sql
SELECT
    user_id,
    COUNT(*) as txn_count,
    SUM(amount_usd) as total_amount,
    GROUP_CONCAT(txn_id) as transaction_ids
FROM transactions
WHERE amount_usd BETWEEN 9000 AND 9999
  AND txn_type IN ('DEPOSIT', 'WITHDRAWAL')
  AND created_at >= datetime('now', '-7 days')
GROUP BY user_id
HAVING txn_count >= 3
ORDER BY txn_count DESC;
```

Chart: Table

---

10. Daily Compliance Summary Report

NL: "Daily compliance report summary"

```
sql

SELECT
    DATE(t.created_at) as date,
    COUNT(*) as total_txns,
    SUM(CASE WHEN t.status = 'FLAGGED' THEN 1 ELSE 0 END) as flagged_txns,
    SUM(CASE WHEN t.amount_usd > 50000 THEN 1 ELSE 0 END) as large_txns,
    ROUND(SUM(t.amount_usd), 2) as total_volume,
    COUNT(DISTINCT t.user_id) as unique_users
FROM transactions t
WHERE t.created_at >= datetime('now', '-30 days')
GROUP BY DATE(t.created_at)
ORDER BY date DESC;
```

Chart: Line Chart (trend) or Table

🔴 SECURITY QUERIES

11. Failed Login Attempts (Last 24 Hours)

NL: "Failed login attempts in last 24 hours"

```
sql

SELECT
    country,
    COUNT(*) as failed_attempts,
    COUNT(DISTINCT ip_address) as unique_ips,
    COUNT(DISTINCT user_id) as affected_users
FROM login_events
WHERE status = 'FAILED'
  AND created_at >= datetime('now', '-24 hours')
GROUP BY country
ORDER BY failed_attempts DESC;
```

Chart: Bar Chart

---

12. Suspicious IPs (Multiple Accounts)

NL: "Suspicious IP addresses with multiple accounts"

```
sql
```

```
SELECT
    ip_address,
    COUNT(DISTINCT user_id) as unique_users,
    COUNT(*) as total_attempts,
    GROUP_CONCAT(DISTINCT user_id) as user_ids,
    MAX(created_at) as last_seen
FROM login_events
WHERE user_id IS NOT NULL
GROUP BY ip_address
HAVING unique_users > 1
ORDER BY unique_users DESC
LIMIT 20;
```

Chart: Table

---

13. Blocked Login Attempts by Country

NL: "Blocked login attempts by country"

```
sql

SELECT
    country,
    COUNT(*) as blocked_count,
    COUNT(DISTINCT ip_address) as unique_ips
FROM login_events
WHERE status = 'BLOCKED'
GROUP BY country
ORDER BY blocked_count DESC;
```

Chart: Bar Chart

---

14. Failed Auth Spike Detection

NL: "Users with more than 5 failed logins in last hour"

```
sql
```

```
SELECT
    user_id,
    email_attempted,
    COUNT(*) as failed_attempts,
    COUNT(DISTINCT ip_address) as unique_ips,
    MIN(created_at) as first_attempt,
    MAX(created_at) as last_attempt
FROM login_events
WHERE status = 'FAILED'
  AND created_at >= datetime('now', '-1 hour')
GROUP BY user_id
HAVING failed_attempts >= 5
ORDER BY failed_attempts DESC;
```

Chart: Table

---

15. New Geographic Logins

NL: "Users logging in from new countries (not seen in 30 days)"

```
sql
```

```
WITH recent_logins AS (
    SELECT DISTINCT user_id, country
    FROM login_events
    WHERE status = 'SUCCESS'
      AND created_at >= datetime('now', '-1 day')
),
historical_countries AS (
    SELECT DISTINCT user_id, country
    FROM login_events
    WHERE status = 'SUCCESS'
      AND created_at < datetime('now', '-1 day')
      AND created_at >= datetime('now', '-30 days')
)
SELECT
    r.user_id,
    r.country as new_country,
    u.full_name
FROM recent_logins r
JOIN users u ON r.user_id = u.user_id
WHERE NOT EXISTS (
    SELECT 1 FROM historical_countries h
    WHERE h.user_id = r.user_id AND h.country = r.country
);
```

Chart: Table

---

16. Off-Hours Trading Activity

NL: "Off-hours trading activity (outside 8AM-8PM)"

```
sql
```

```
SELECT
    user_id,
    COUNT(*) as off_hours_txns,
    SUM(amount_usd) as total_volume,
    GROUP_CONCAT(DISTINCT strftime('%H', created_at)) as hours_active
FROM transactions
WHERE txn_type = 'TRADE'
  AND (CAST(strftime('%H', created_at) AS INTEGER) < 8
       OR CAST(strftime('%H', created_at) AS INTEGER) >= 20)
  AND created_at >= datetime('now', '-7 days')
GROUP BY user_id
HAVING off_hours_txns >= 3
ORDER BY off_hours_txns DESC;
```

Chart: Table

---

17. Device Type Distribution

NL: "Login attempts by device type"

```sql
SELECT
    device_type,
    COUNT(*) as total_attempts,
    SUM(CASE WHEN status = 'SUCCESS' THEN 1 ELSE 0 END) as successful,
    SUM(CASE WHEN status = 'FAILED' THEN 1 ELSE 0 END) as failed,
    ROUND(SUM(CASE WHEN status = 'SUCCESS' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 1) as success_rate
FROM login_events
GROUP BY device_type
ORDER BY total_attempts DESC;
```

Chart: Pie Chart

---

18. Account Takeover Risk Indicators

NL: "Account takeover risk indicators"

```sql
```

```
SELECT
  l.user_id,
  u.full_name,
  u.email,
  COUNT(DISTINCT l.ip_address) as unique_ips_24h,
  COUNT(DISTINCT l.country) as unique_countries_24h,
  COUNT(DISTINCT l.device_fingerprint) as unique_devices_24h,
  SUM(CASE WHEN l.status = 'FAILED' THEN 1 ELSE 0 END) as failed_attempts
FROM login_events l
JOIN users u ON l.user_id = u.user_id
WHERE l.created_at >= datetime('now', '-24 hours')
GROUP BY l.user_id
HAVING unique_ips_24h > 3 OR unique_countries_24h > 1 OR failed_attempts > 3
ORDER BY unique_ips_24h DESC;
```

Chart: Table

---

19. Open Security Alerts by Severity

NL: "Open security alerts by severity"

```sql
SELECT
  severity,
  COUNT(*) as alert_count,
  GROUP_CONCAT(DISTINCT rule_name) as rules_triggered
FROM alerts
WHERE status IN ('OPEN', 'INVESTIGATING')
GROUP BY severity
ORDER BY
  CASE severity
    WHEN 'CRITICAL' THEN 1
    WHEN 'HIGH' THEN 2
    WHEN 'MEDIUM' THEN 3
    ELSE 4
  END;
```

Chart: Bar Chart

---

20. Velocity Alert Detection

NL: "Users exceeding trade velocity limits (>20 trades in 5 min)"

```sql
SELECT
    user_id,
    COUNT(*) as trade_count,
    MIN(created_at) as window_start,
    MAX(created_at) as window_end,
    SUM(amount_usd) as total_volume
FROM transactions
WHERE txn_type = 'TRADE'
  AND created_at >= datetime('now', '-1 hour')
GROUP BY user_id, strftime('%Y-%m-%d %H', created_at), CAST(strftime('%M', created_at) AS INTEGER) / 5
HAVING trade_count > 20
ORDER BY trade_count DESC;
```

Chart: Table

🟣 OPERATIONS QUERIES

21. Daily Transaction Volume Trend

NL: "Daily transaction volume trend"

```sql
SELECT
    DATE(created_at) as date,
    COUNT(*) as txn_count,
    ROUND(SUM(amount_usd), 2) as total_volume,
    ROUND(AVG(amount_usd), 2) as avg_amount
FROM transactions
WHERE created_at >= datetime('now', '-30 days')
GROUP BY DATE(created_at)
ORDER BY date ASC;
```

Chart: Line Chart / Area Chart

---

22. Top 10 Users by Transaction Volume

NL: "Top 10 users by transaction volume"

```sql
SELECT
    t.user_id,
    u.full_name,
    u.country,
    u.risk_level,
    COUNT(t.txn_id) as txn_count,
    ROUND(SUM(t.amount_usd), 2) as total_volume,
    ROUND(AVG(t.amount_usd), 2) as avg_txn
FROM transactions t
JOIN users u ON t.user_id = u.user_id
GROUP BY t.user_id
ORDER BY total_volume DESC
LIMIT 10;
```

Chart: Bar Chart

---

23. Transaction Breakdown by Type

NL: "Transaction breakdown by type"

```sql
SELECT
    txn_type,
    COUNT(*) as count,
    ROUND(SUM(amount_usd), 2) as total_volume,
    ROUND(AVG(amount_usd), 2) as avg_amount,
    ROUND(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM transactions), 1) as percentage
FROM transactions
GROUP BY txn_type
ORDER BY total_volume DESC;
```

Chart: Pie Chart

---

24. Transaction Success Rate

NL: "Transaction success rate by type"

```sql
```

```
SELECT
    txn_type,
    COUNT(*) as total,
    SUM(CASE WHEN status = 'COMPLETED' THEN 1 ELSE 0 END) as completed,
    SUM(CASE WHEN status = 'FAILED' THEN 1 ELSE 0 END) as failed,
    SUM(CASE WHEN status = 'PENDING' THEN 1 ELSE 0 END) as pending,
    ROUND(SUM(CASE WHEN status = 'COMPLETED' THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 1) as success_rate
FROM transactions
GROUP BY txn_type
ORDER BY total DESC;
```

Chart: Table / Bar Chart

---

25. Users by Country

NL: "User distribution by country"

```sql
SELECT
    country,
    COUNT(*) as user_count,
    SUM(CASE WHEN risk_level = 'HIGH' THEN 1 ELSE 0 END) as high_risk,
    SUM(CASE WHEN kyc_status = 'VERIFIED' THEN 1 ELSE 0 END) as verified,
    ROUND(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM users), 1) as percentage
FROM users
GROUP BY country
ORDER BY user_count DESC;
```

Chart: Bar Chart

---

26. Revenue by Instrument

NL: "Trading volume breakdown by instrument"

```sql
```

```
SELECT
    instrument,
    COUNT(*) as trade_count,
    ROUND(SUM(amount_usd), 2) as total_volume,
    ROUND(AVG(amount_usd), 2) as avg_trade
FROM transactions
WHERE txn_type = 'TRADE' AND instrument IS NOT NULL
GROUP BY instrument
ORDER BY total_volume DESC;
```

Chart: Bar Chart / Pie Chart

---

27. Peak Trading Hours Analysis

NL: "Peak trading hours analysis"

```sql
SELECT
    CAST(strftime('%H', created_at) AS INTEGER) as hour,
    COUNT(*) as txn_count,
    ROUND(SUM(amount_usd), 2) as total_volume
FROM transactions
WHERE txn_type = 'TRADE'
GROUP BY hour
ORDER BY hour;
```

Chart: Bar Chart (Histogram)

---

28. User Growth Trend

NL: "User registration trend by month"

```sql
```

```
SELECT
    strftime('%Y-%m', created_at) as month,
    COUNT(*) as new_users,
    SUM(COUNT(*)) OVER (ORDER BY strftime('%Y-%m', created_at)) as cumulative_users
FROM users
GROUP BY month
ORDER BY month;
```

Chart: Line Chart

---

29. Average Transaction Size by Risk Level

NL: "Average transaction size by user risk level"

```sql
SELECT
    u.risk_level,
    COUNT(t.txn_id) as txn_count,
    ROUND(AVG(t.amount_usd), 2) as avg_amount,
    ROUND(MIN(t.amount_usd), 2) as min_amount,
    ROUND(MAX(t.amount_usd), 2) as max_amount
FROM users u
JOIN transactions t ON u.user_id = t.user_id
GROUP BY u.risk_level
ORDER BY avg_amount DESC;
```

Chart: Bar Chart

---

30. Failed Transaction Analysis

NL: "Failed transaction root cause analysis"

```sql
```

```
SELECT
    txn_type,
    payment_method,
    COUNT(*) as failed_count,
    ROUND(SUM(amount_usd), 2) as failed_volume
FROM transactions
WHERE status = 'FAILED'
GROUP BY txn_type, payment_method
ORDER BY failed_count DESC;
```

Chart: Table

🟠 ALERT QUERIES

31. Alert Distribution by Rule

NL: "Alerts grouped by detection rule"

```sql
SELECT
    rule_name,
    severity,
    COUNT(*) as total_alerts,
    SUM(CASE WHEN status = 'OPEN' THEN 1 ELSE 0 END) as open,
    SUM(CASE WHEN status = 'RESOLVED' THEN 1 ELSE 0 END) as resolved
FROM alerts
GROUP BY rule_name, severity
ORDER BY total_alerts DESC;
```

Chart: Bar Chart

---

32. Alert Resolution Time

NL: "Average alert resolution time by severity"

```sql
```

```
SELECT
    severity,
    COUNT(*) as resolved_count,
    ROUND(AVG(JULIANDAY(resolved_at) - JULIANDAY(created_at)) * 24, 1) as avg_hours_to_resolve
FROM alerts
WHERE status = 'RESOLVED' AND resolved_at IS NOT NULL
GROUP BY severity
ORDER BY
    CASE severity
        WHEN 'CRITICAL' THEN 1
        WHEN 'HIGH' THEN 2
        WHEN 'MEDIUM' THEN 3
        ELSE 4
    END;
```

Chart: Bar Chart

---

33. Users with Most Alerts

NL: "Users with most open alerts"

```sql
SELECT
    a.user_id,
    u.full_name,
    u.risk_level,
    COUNT(*) as alert_count,
    GROUP_CONCAT(DISTINCT a.rule_name) as triggered_rules
FROM alerts a
JOIN users u ON a.user_id = u.user_id
WHERE a.status IN ('OPEN', 'INVESTIGATING')
GROUP BY a.user_id
ORDER BY alert_count DESC
LIMIT 15;
```

Chart: Table

---

34. Alert Trend Over Time

NL: "Alert trend over the last 30 days"

```sql
SELECT
    DATE(created_at) as date,
    COUNT(*) as total_alerts,
    SUM(CASE WHEN severity = 'CRITICAL' THEN 1 ELSE 0 END) as critical,
    SUM(CASE WHEN severity = 'HIGH' THEN 1 ELSE 0 END) as high,
    SUM(CASE WHEN severity = 'MEDIUM' THEN 1 ELSE 0 END) as medium,
    SUM(CASE WHEN severity = 'LOW' THEN 1 ELSE 0 END) as low
FROM alerts
WHERE created_at >= datetime('now', '-30 days')
GROUP BY DATE(created_at)
ORDER BY date;
```

Chart: Stacked Area Chart / Line Chart

---

35. Unassigned Critical Alerts

NL: "Show unassigned critical and high severity alerts"

```sql
SELECT
    a.alert_id,
    a.rule_name,
    a.severity,
    a.user_id,
    u.full_name,
    a.created_at,
    ROUND(JULIANDAY('now') - JULIANDAY(a.created_at), 1) as days_open
FROM alerts a
LEFT JOIN users u ON a.user_id = u.user_id
WHERE a.status = 'OPEN'
  AND a.assigned_to IS NULL
  AND a.severity IN ('CRITICAL', 'HIGH')
ORDER BY
    CASE a.severity WHEN 'CRITICAL' THEN 1 ELSE 2 END,
    a.created_at ASC;
```

Chart: Table

📊 DASHBOARD KPI QUERIES

36. Executive Dashboard KPIs

NL: "Show me the main KPIs"

```sql
-- Total Volume (Today)
SELECT 'Today Volume' as metric, ROUND(SUM(amount_usd), 2) as value
FROM transactions WHERE DATE(created_at) = DATE('now')
UNION ALL
-- Active Users (Today)
SELECT 'Active Users', COUNT(DISTINCT user_id)
FROM transactions WHERE DATE(created_at) = DATE('now')
UNION ALL
-- Open Alerts
SELECT 'Open Alerts', COUNT(*)
FROM alerts WHERE status = 'OPEN'
UNION ALL
-- Flagged Transactions (Today)
SELECT 'Flagged Today', COUNT(*)
FROM transactions WHERE status = 'FLAGGED' AND DATE(created_at) = DATE('now')
UNION ALL
-- Pending KYC
SELECT 'Pending KYC', COUNT(*)
FROM users WHERE kyc_status = 'PENDING'
UNION ALL
-- High Risk Users
SELECT 'High Risk Users', COUNT(*)
FROM users WHERE risk_level = 'HIGH';
```

Chart: KPI Cards

🔍 SEARCH/LOOKUP QUERIES

37. Search User by Email or Name

NL: "Find user john.smith"

```sql
```

```
SELECT *
FROM users
WHERE email LIKE '%john.smith%'
  OR full_name LIKE '%john%smith%'
  OR user_id LIKE '%john%';
```

Chart: Table

---

38. Transaction History for User

NL: "Show transaction history for user USR-0042"

```sql
SELECT
    txn_id,
    txn_type,
    instrument,
    amount_usd,
    status,
    created_at
FROM transactions
WHERE user_id = 'USR-0042'
ORDER BY created_at DESC
LIMIT 50;
```

Chart: Table

---

39. User Activity Summary

NL: "Complete activity summary for user USR-0042"

```sql
```

```sql
SELECT
    'Profile' as category,
    u.full_name || ' | ' || u.country || ' | ' || u.risk_level || ' | KYC: ' || u.kyc_status as details
FROM users u WHERE u.user_id = 'USR-0042'
UNION ALL
SELECT 'Total Transactions', COUNT(*) || ' transactions ($' || ROUND(SUM(amount_usd), 2) || ' volume)'
FROM transactions WHERE user_id = 'USR-0042'
UNION ALL
SELECT 'Open Alerts', COUNT(*) || ' alerts'
FROM alerts WHERE user_id = 'USR-0042' AND status = 'OPEN'
UNION ALL
SELECT 'Login Attempts (30d)', COUNT(*) || ' attempts (' || SUM(CASE WHEN status='FAILED' THEN 1 ELSE 0 END) ||
FROM login_events WHERE user_id = 'USR-0042' AND created_at >= datetime('now', '-30 days');
```

Chart: Table

---

40. Recent Activity Feed

NL: "Show recent platform activity"

```
sql
```

```
SELECT
    'Transaction' as type,
    txn_id as id,
    user_id,
    txn_type || ' $' || amount_usd as description,
    status,
    created_at
FROM transactions
WHERE created_at >= datetime('now', '-1 hour')
UNION ALL
SELECT
    'Alert' as type,
    alert_id as id,
    user_id,
    rule_name || ' (' || severity || ')' as description,
    status,
    created_at
FROM alerts
WHERE created_at >= datetime('now', '-1 hour')
UNION ALL
SELECT
    'Login' as type,
    event_id as id,
    user_id,
    status || ' from ' || country as description,
    status,
    created_at
FROM login_events
WHERE created_at >= datetime('now', '-1 hour')
ORDER BY created_at DESC
LIMIT 50;
```

Chart: Activity Feed / Table

📝 NOTES FOR NL2SQL MAPPING

Query Pattern Recognition:

NL Pattern	SQL Pattern
"show/list/get all"	SELECT *

NL Pattern	SQL Pattern
"how many" / "count"	SELECT COUNT(*)
"total" / "sum"	SELECT SUM()
"average"	SELECT AVG()
"top N" / "highest"	ORDER BY ... DESC LIMIT N
"last N days"	WHERE created_at >= datetime('now', '-N days')
"by country/type/status"	GROUP BY column
"between X and Y"	WHERE column BETWEEN X AND Y
"more than" / "greater than"	WHERE column > value
"pending/flagged/high-risk"	WHERE status = 'VALUE'

Time Filters:

NL	SQL
"today"	DATE(created_at) = DATE('now')
"yesterday"	DATE(created_at) = DATE('now', '-1 day')
"this week"	created_at >= datetime('now', '-7 days')
"this month"	created_at >= datetime('now', '-30 days')
"last 24 hours"	created_at >= datetime('now', '-24 hours')

End of Query Reference Guide