1: Preview data to analyze format

```sql
SELECT *
FROM `lepidge-project.fraud_analytics.bank_transactions`
LIMIT 10;
```

2: Check for nulls

```sql
SELECT
 COUNT(*) AS total_rows,
 COUNTIF(transactionID IS NULL) AS null_transactionID,
 COUNTIF(AccountID IS NULL) AS null_AccountID,
 COUNTIF(TransactionAmount IS NULL) AS null_TransactionAmount,
 COUNTIF(TransactionDate IS NULL) AS null_TransactionDate,
 COUNTIF(TransactionType IS NULL) AS null_TransactionType,
 COUNTIF(Location IS NULL) AS null_Location,
 COUNTIF(DeviceID IS NULL) AS null_DeviceID,
 COUNTIF(`IP Address` IS NULL) AS null_IPAddress,
 COUNTIF(MerchantID IS NULL) AS null_MerchantID,
 COUNTIF(Channel IS NULL) AS null_Channel,
 COUNTIF(CustomerAge IS NULL) AS null_CustomerAge,
 COUNTIF(CustomerOccupation IS NULL) AS null_CustomerOccupation,
 COUNTIF(TransactionDuration IS NULL) AS null_TransactionDuration,
 COUNTIF(LoginAttempts IS NULL) AS null_LoginAttempts,
 COUNTIF(AccountBalance IS NULL) AS null_AccountBalance,
 COUNTIF(PreviousTransactionDate IS NULL) AS null_PreviousTransactionDate
FROM `lepidge-project.fraud_analytics.bank_transactions`;
```

3: Search for abnormalities

```sql
SELECT
 COUNT(*) AS total_transactions,
 COUNTIF(TransactionAmount > 1000) AS high_value_transactions,
 AVG(TransactionAmount) AS avg_amount,
 MAX(TransactionAmount) AS max_amount
FROM `lepidge-project.fraud_analytics.bank_transactions`;
```

4: Explore categorical patterns

```
SELECT
 TransactionType,
 COUNT(*) AS count,
 ROUND(100 * COUNT(*) / (SELECT COUNT(*) FROM
`lepidge-project.fraud_analytics.bank_transactions`), 2) AS percent
FROM `lepidge-project.fraud_analytics.bank_transactions`
GROUP BY TransactionType
ORDER BY count DESC;
```

Row	TransactionType	count	percent
1	Debit	1944	77.39
2	Credit	568	22.61

5: Explore Channels

```
SELECT
 Channel,
 COUNT(*) AS count,
 ROUND(100 * COUNT(*) / (SELECT COUNT(*) FROM
`lepidge-project.fraud_analytics.bank_transactions`), 2) AS percent
FROM `lepidge-project.fraud_analytics.bank_transactions`
GROUP BY Channel
ORDER BY count DESC;
```

Row	Channel	count	percent
1	Branch	868	34.55
2	ATM	833	33.16
3	Online	811	32.29

6: Explore Locations

```
SELECT
 Location,
 COUNT(*) AS count
FROM `lepidge-project.fraud_analytics.bank_transactions`
GROUP BY Location
ORDER BY count DESC
LIMIT 10;
```

Row	Location	count
1	Fort Worth	70
2	Los Angeles	69
3	Charlotte	68
4	Oklahoma City	68
5	Philadelphia	67
6	Tucson	67
7	Omaha	65
8	Miami	64

7: Built SQL query that adds fraud indicator flags to dataset (higher fraud score of 3 or 4 means more likely to be fraud)

```
SELECT *,
 CASE WHEN TransactionAmount > 1000 THEN 1 ELSE 0 END AS flag_high_value,
 CASE WHEN LoginAttempts > 3 THEN 1 ELSE 0 END AS flag_excessive_logins,
 CASE WHEN CustomerAge < 25 AND TransactionAmount > 750 THEN 1 ELSE 0 END AS
flag_young_big_spender,
```

```
 CASE WHEN Channel IN ('ATM', 'Online') THEN 1 ELSE 0 END AS flag_risky_channel,
 (
   CASE WHEN TransactionAmount > 1000 THEN 1 ELSE 0 END +
   CASE WHEN LoginAttempts > 3 THEN 1 ELSE 0 END +
   CASE WHEN CustomerAge < 25 AND TransactionAmount > 750 THEN 1 ELSE 0 END +
   CASE WHEN Channel IN ('ATM', 'Online') THEN 1 ELSE 0 END
 ) AS fraud_score

FROM `lepidge-project.fraud_analytics.bank_transactions`
```

Row	TransactionID	AccountID	TransactionAmou...	TransactionDate	TransactionTyp
1	TX000006	AC00393	92.15	2023-04-03 17:15:01 UTC	Debit
2	TX000143	AC00163	227.14	2023-07-03 17:42:08 UTC	Debit
3	TX000254	AC00442	218.96	2023-08-30 16:11:47 UTC	Debit
4	TX000413	AC00421	242.39	2023-11-20 16:29:28 UTC	Credit
5	TX000463	AC00074	17.45	2023-09-13 16:41:19 UTC	Debit
6	TX000476	AC00464	1431.3	2023-09-04 18:46:40 UTC	Debit
7	TX000481	AC00306	144.3	2023-12-08 17:24:44 UTC	Debit
8	TX000518	AC00122	358.18	2023-04-10 16:44:52 UTC	Debit

8: Analyzed scores based on how many risky transactions exist

```
 SELECT `fraud_score` AS `fraud_score`, `count` AS `count`, `percent` AS `percent`
 FROM (
SELECT
 fraud_score,
 COUNT(*) AS count,
 ROUND(100 * COUNT(*) / (SELECT COUNT(*) FROM
`lepidge-project.fraud_analytics.bank_transactions`), 2) AS percent
FROM (
 SELECT *,
   (
     CASE WHEN TransactionAmount > 1000 THEN 1 ELSE 0 END +
     CASE WHEN LoginAttempts > 3 THEN 1 ELSE 0 END +
     CASE WHEN CustomerAge < 25 AND TransactionAmount > 750 THEN 1 ELSE 0 END +
     CASE WHEN Channel IN ('ATM', 'Online') THEN 1 ELSE 0 END
```

```
   ) AS fraud_score
 FROM `lepidge-project.fraud_analytics.bank_transactions`
)
GROUP BY fraud_score
ORDER BY fraud_score DESC
)
 LIMIT 500
```

Row	fraud_score	count	percent
1	4	1	0.04
2	3	12	0.48
3	2	110	4.38
4	1	1576	62.74
5	0	813	32.36