1. Database Schema Setup

We created foundational tables to store lab batches, sample metadata, assay methods, assay results, and QA/QC flags.

```
CREATE TABLE LabBatches (
    BatchID INT PRIMARY KEY,
    BatchDate DATE,
    LabTechnician NVARCHAR(100)
);

CREATE TABLE Samples (
    SampleID INT PRIMARY KEY,
    BatchID INT FOREIGN KEY REFERENCES LabBatches(BatchID),
    SampleType NVARCHAR(50),
    RockType NVARCHAR(50),
    WeightAnalyzed_grams FLOAT
);

CREATE TABLE AssayMethods (
    MethodID INT PRIMARY KEY,
    Element NVARCHAR(10),
    DetectionLimit_ppm FLOAT
);

CREATE TABLE AssayResults (
    ResultID INT PRIMARY KEY,
    SampleID INT FOREIGN KEY REFERENCES Samples(SampleID),
    MethodID INT FOREIGN KEY REFERENCES AssayMethods(MethodID),
    Concentration_ppm FLOAT
);

CREATE TABLE QAChecks (
    SampleID INT FOREIGN KEY REFERENCES Samples(SampleID),
    IsDuplicate BIT,
    IsBlank BIT,
    IsStandard BIT
);
```

2. Synthetic Data Generation

We inserted 100 samples across 2 batches, with randomized weights and sample types (Rock, Duplicate, Blank, Standard). Standards received lower weights to simulate analytical precision.

```
DECLARE @i INT = 1;
WHILE @i <= 100
BEGIN
    INSERT INTO Samples (SampleID, BatchID, SampleType, RockType,
WeightAnalyzed_grams)
    VALUES (
        @i,
        CASE WHEN @i <= 50 THEN 1 ELSE 2 END,
        CASE
```

```
            WHEN @i % 20 = 0 THEN 'Standard'
            WHEN @i % 15 = 0 THEN 'Blank'
            WHEN @i % 10 = 0 THEN 'Duplicate'
            ELSE 'Rock'
        END,
        'Granite',
        ROUND(
            CASE
                WHEN @i % 20 = 0 THEN RAND() * 0.01 + 0.05
                ELSE RAND() * 0.25 + 2.00
            END,
            3
        )
    );
    SET @i = @i + 1;
END;
```

3. Metal Content Calculation

We calculated metal content in milligrams by multiplying concentration (ppm) by weight analyzed (grams).

```
SELECT
    s.SampleCode,
    s.SampleType,
    s.Location,
    s.BatchID,
    ar.Element,
    ar.Concentration_ppm,
    ar.WeightAnalyzed_grams,
    ROUND(ar.Concentration_ppm * ar.WeightAnalyzed_grams, 3) AS MetalContent_mg
FROM AssayResults ar
JOIN Samples s ON ar.SampleID = s.SampleID
ORDER BY s.SampleCode, ar.Element;
```

	SampleCode	SampleType	Location	BatchID	Element	Concentration_ppm	WeightAnalyzed_grams	MetalContent_mg
1	SMP-001	Rock	Zone B	1	Ag	4.161	2.148	8.938
2	SMP-001	Rock	Zone B	1	Zn	4.538	1.804	8.187
3	SMP-002	Rock	Zone C	1	Au	1.291	1.787	2.307
4	SMP-004	Rock	Zone B	1	Ag	1.75	1.778	3.111

4. Total Metal Content by Batch

This query aggregates total metal content per lab batch.

```
SELECT S.BatchID,
    SUM(AR.Concentration_ppm * AR.WeightAnalyzed_grams) AS TotalMetalContent_mg
FROM AssayResults AR
JOIN Samples S ON AR.SampleID = S.SampleID
GROUP BY S.BatchID;
```

	BatchID	TotalMetalContent_mg
1	1	498.533492
2	2	486.050446

5. Total Metal Content by Element

This query shows total metal recovery for each element across all samples.

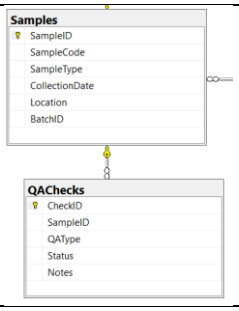
```
SELECT [Element],
    SUM(Concentration_ppm * WeightAnalyzed_grams) AS TotalMetalContent_mg
FROM AssayResults

GROUP BY [Element];
```

	Element	TotalMetalContent_mg
1	Ag	110.088388
2	Au	195.56076
3	Cu	151.168001
4	Pb	116.260113
5	Zn	411.506676

6. QA/QC Sample Flagging

We flagged samples as duplicates, blanks, or standards for quality control tracking.

```
INSERT INTO QAChecks (SampleID, QAType, Status, Notes)
SELECT
    SampleID,
    SampleType,          -- becomes QAType
    'Pending',           -- default status
    NULL                 -- no notes yet
FROM Samples
WHERE SampleType IN ('Duplicate', 'Blank', 'Standard');
```

This says: "Get data from the `Samples` table and add some rows to `QAChecks table`. For each sample: ✓ Use its `SampleID` ✓ Use its SampleType as the `QAType` ✓ Set the `Status` to 'Pending' ✓ Leave `Notes` empty (NULL)"	
`WHERE SampleType IN ('Duplicate', 'Blank', 'Standard');`	This filters the samples. Only include samples that are **QA types**:`'Duplicate', 'Blank', 'Standard'`

	CheckID	SampleID	QAType	Status	Notes
5	24	156	Blank	Pass	Auto-generated QA check
6	25	166	Duplicate	Pass	Auto-generated QA check
7	26	181	Blank	Pass	Auto-generated QA check
8	27	186	Duplicate	Pass	Auto-generated QA check
9	28	196	Standard	Pass	Auto-generated QA check
10	29	206	Duplicate	Pass	Auto-generated QA check
11	30	126	Duplicate	Pending	NULL
12	31	131	Blank	Pending	NULL
13	32	136	Standard	Pending	NULL
14	33	146	Duplicate	Pending	NULL
15	34	156	Blank	Pending	NULL
16	35	166	Duplicate	Pending	NULL

Step 7: Create a Summary View

Purpose:

To make it easy to query and visualize key assay data — including:

- Sample info
- Elemental concentrations
- Analytical weights
- Metal content
- QA type (if any)

Create a View

```
CREATE VIEW AssaySummary AS
SELECT
    s.SampleID,
    s.SampleType,
    s.BatchID,
    ar.WeightAnalyzed_grams,
    ar.Element,
    ar.Concentration_ppm,
    ROUND(ar.Concentration_ppm * ar.WeightAnalyzed_grams, 3) AS MetalContent_mg,
    qc.QAType,
    qc.Status
FROM Samples s
JOIN AssayResults ar ON s.SampleID = ar.SampleID
LEFT JOIN QAChecks qc ON s.SampleID = qc.SampleID;


SELECT * FROM AssaySummary WHERE QAType IS NOT NULL;
```

	SampleID	SampleType	BatchID	WeightAnalyzed_grams	Element	Concentration_ppm	MetalContent_mg	QAType	Status
11	181	Blank	2	2.030	Au	2.88	5.846	Blank	Pass
12	181	Blank	2	2.030	Au	2.88	5.846	Blank	Pending
13	126	Duplicate	1	2.157	Zn	1.659	3.578	Duplicate	Pass
14	126	Duplicate	1	2.157	Zn	1.659	3.578	Duplicate	Pending
15	206	Duplicate	2	2.191	Au	3.002	6.577	Duplicate	Pass
16	206	Duplicate	2	2.191	Au	3.002	6.577	Duplicate	Pending
17	146	Duplicate	1	2.204	Au	2.925	6.447	Duplicate	Pass

OR

```
SELECT BatchID, Element, AVG(MetalContent_mg) AS AvgMetalYield
FROM AssaySummary
GROUP BY BatchID, Element;
```

	BatchID	Element	AvgMetalYield
1	1	Ag	4.34816666666667
2	2	Ag	5.3035
3	1	Au	4.85526315789474
4	2	Au	5.77259090909091
5	1	Cu	5.43554545454545
6	2	Cu	4.37273913043478
7	1	Pb	4.2302

Step 8: Add Detection Limits

Purpose:

To identify results below the minimum reliable concentration for each element.

Step 8.1: Create a `DetectionLimits` Table

```
CREATE TABLE DetectionLimits (
    Element NVARCHAR(10) PRIMARY KEY,
    DetectionLimit_ppm REAL);
```

Example Data

```
INSERT INTO DetectionLimits (Element, DetectionLimit_ppm)
VALUES
      ('Au', 0.005),
      ('Ag', 0.1),
      ('Cu', 1.0),
      ('Pb', 2.0);
```

Step 8.2: Update the `AssaySummary` View

Now we'll join with `DetectionLimits` and flag results below detection.

Fix: Add `DetectionStatus` as a Computed Column

Here's how you might define it in your view:

```
SELECT
    a.SampleID,
    a.Element,
    a.Concentration_ppm,
    d.DetectionLimit_ppm,
    CASE
        WHEN a.Concentration_ppm >= d.DetectionLimit_ppm THEN 'Detected'
        ELSE 'Below Detection'
    END AS DetectionStatus
FROM AssayResults a

JOIN DetectionLimits d ON a.Element = d.Element;
```

This ensures `DetectionStatus` is part of the result set.

Now Create another **VIEW**

```
CREATE VIEW AssaySummary2 AS
SELECT
    s.SampleID,
    s.SampleType,
    s.BatchID,
    ar.Element,
    ar.WeightAnalyzed_grams,
    ar.Concentration_ppm,
    ROUND(ar.Concentration_ppm * ar.WeightAnalyzed_grams, 3) AS MetalContent_mg,
    dl.DetectionLimit_ppm,
    CASE
        WHEN ar.Concentration_ppm < dl.DetectionLimit_ppm THEN 'Below Detection'
        ELSE 'Detected'
    END AS DetectionStatus,
    qc.QAType,
    qc.Status
FROM Samples s
JOIN AssayResults ar ON s.SampleID = ar.SampleID
LEFT JOIN QAChecks qc ON s.SampleID = qc.SampleID
LEFT JOIN DetectionLimits dl ON ar.Element = dl.Element;
```

Sample Query: Flag All Below Detection

```
SELECT * FROM AssaySummary2 WHERE DetectionStatus = 'Below Detection';
```

	SampleID	SampleType	BatchID	Element	WeightAnalyzed_grams	Concentration_ppm	MetalContent_mg	DetectionLimit_ppm	DetectionStatus	QAType	Status
1	171	Rock	2	Cu	1.787	0.196	0.35	1	Below Detection	NULL	NULL
2	195	Rock	2	Pb	1.991	0.161	0.321	2	Below Detection	NULL	NULL
3	169	Rock	2	Pb	2.165	0.608	1.316	2	Below Detection	NULL	NULL
4	147	Rock	1	Pb	1.894	0.598	1.133	2	Below Detection	NULL	NULL
5	134	Rock	1	Pb	2.218	0.967	2.145	2	Below Detection	NULL	NULL
6	143	Rock	1	Cu	2.110	0.221	0.466	1	Below Detection	NULL	NULL
7	184	Rock	2	Pb	2.217	0.686	1.521	2	Below Detection	NULL	NULL
8	115	Rock	1	Cu	1.781	0.865	1.541	1	Below Detection	NULL	NULL

Count how many times each **element** was reported as **"Below Detection"** in the `AssaySummary2` table. It helps identify which elements are frequently undetected.

```sql
SELECT Element, COUNT(*) AS BelowDetectionCount
FROM AssaySummary2
WHERE DetectionStatus = 'Below Detection'
GROUP BY Element;
```

	Element	BelowDetectionCount
1	Cu	7
2	Pb	12

Number of QA samples

```sql
SELECT QAType, DetectionStatus, COUNT(*) AS Count
FROM AssaySummary2
WHERE QAType IS NOT NULL
GROUP BY QAType, DetectionStatus;
```

	QAType	DetectionStatus	Count
1	Blank	Below Detection	2
2	Duplicate	Below Detection	2
3	Blank	Detected	8
4	Duplicate	Detected	22
5	Standard	Detected	6

To evaluate the performance of QA samples and spot any issues in assay accuracy or precision.
That SQL query counts how many QA samples (like **duplicates**, **blanks**, or **standards**) fall into each **detection status** (e.g., "Detected", "Below Detection").

Calculate total and average `MetalContent_mg`

```sql
SELECT BatchID, Element,
       ROUND(SUM(MetalContent_mg),2) AS TotalYield,
       ROUND(AVG(MetalContent_mg),2) AS AvgYield
FROM AssaySummary2
GROUP BY BatchID, Element;
```

	BatchID	Element	TotalYield	AvgYield
1	1	Ag	78.27	4.35
2	2	Ag	31.82	5.3
3	1	Au	92.25	4.86
4	2	Au	127	5.77
5	1	Cu	59.79	5.44
6	2	Cu	100.57	4.37
7	1	Pb	42.3	4.23
8	2	Pb	77.85	5.19
9	1	Zn	257.93	4.69
10	2	Zn	183.64	4.48

This query calculates:
-Total metal yield per batch and element
-Average metal content per sample in each batch
It helps you:
-Compare batch performance
-Spot inconsistencies or low-yield batches
-Track which elements are contributing most to recovery

Filter By Gold

```sql
SELECT BatchID,
```

```
       ROUND(SUM(MetalContent_mg),2) AS TotalGoldYield,
       ROUND(AVG(MetalContent_mg),2) AS AvgGoldYield
FROM AssaySummary2
WHERE Element = 'Au'
GROUP BY BatchID;

Sometimes add the order:
ORDER BY TotalGoldYield DESC;
```

	BatchID	TotalGoldYield	AvgGoldYield
1	1	92.25	4.86
2	2	127	5.77

- Focuses only on gold assays
- Shows total and average gold content per batch
- Useful for tracking gold recovery and batch quality

```
SELECT TOP 5
    BatchID,
    COUNT(*) AS SampleCount,
    SUM(MetalContent_mg) AS TotalGoldYield,
    AVG(MetalContent_mg) AS AvgGoldYield
FROM AssaySummary2
WHERE Element = 'Au'
GROUP BY BatchID
ORDER BY TotalGoldYield DESC;
```

	BatchID	SampleCount	TotalGoldYield	AvgGoldYield
1	2	22	126.997	5.77259090909091
2	1	19	92.25	4.85526315789474

- Filters for gold (Au)
- Calculates total and average yield
- Counts samples per batch
- Sorts by highest yield
- Limits to top 5 batches

```
SELECT BatchID,
  SUM(CASE WHEN Element = 'Au' THEN MetalContent_mg ELSE 0 END) AS TotalGoldYield,
  AVG(CASE WHEN Element = 'Au' THEN MetalContent_mg ELSE NULL END) AS AvgGoldYield,
  SUM(CASE WHEN Element = 'Ag' THEN MetalContent_mg ELSE 0 END) AS TotalSilverYield,
  AVG(CASE WHEN Element = 'Ag' THEN MetalContent_mg ELSE NULL END) AS AvgSilverYield
FROM AssaySummary2
WHERE Element IN ('Au', 'Ag')
GROUP BY BatchID
ORDER BY TotalGoldYield DESC;
```

	BatchID	TotalGoldYield	AvgGoldYield	TotalSilverYield	AvgSilverYield
1	2	126.997	5.77259090909091	31.821	5.3035
2	1	92.25	4.85526315789474	78.267	4.34816666666667

- Separates gold and silver using CASE statements.
- Calculates total and average yields for each metal.
- Groups by batch so you can compare them side by side.
- Sorts by gold yield, but you can easily switch to silver if needed.

```
SELECT BatchID,
 SUM(CASE WHEN Element = 'Au' THEN MetalContent_mg ELSE 0 END) AS TotalGoldYield,
 AVG(CASE WHEN Element = 'Au' THEN MetalContent_mg ELSE NULL END) AS AvgGoldYield,
```

```
 SUM(CASE WHEN Element = 'Ag' THEN MetalContent_mg ELSE 0 END) AS TotalSilverYield,
 AVG(CASE WHEN Element = 'Ag' THEN MetalContent_mg ELSE NULL END) AS AvgSilverYield,
 SUM(CASE WHEN Element = 'Cu' THEN MetalContent_mg ELSE 0 END) AS TotalCopperYield,
 AVG(CASE WHEN Element = 'Cu' THEN MetalContent_mg ELSE NULL END) AS AvgCopperYield
FROM AssaySummary2
WHERE Element IN ('Au', 'Ag', 'Cu')
GROUP BY BatchID
ORDER BY TotalGoldYield DESC;
```

	BatchID	TotalGoldYield	AvgGoldYield	TotalSilverYield	AvgSilverYield	TotalCopperYield	AvgCopperYield
1	2	126.997	5.77259090909091	31.821	5.3035	100.573	4.37273913043478
2	1	92.25	4.85526315789474	78.267	4.34816666666667	59.791	5.43554545454545

Gold, Silver, and Copper Yield by Batch

- Each batch has three bars: Au, Ag, Cu.
- You can easily spot which metal dominates in each batch.

Metal Yield Report by Batch

- **Summary:**
- This report presents total and average metal yields (in milligrams) for Gold, Silver, and Copper across all assay batches. Batches are sorted by **total gold yield** in descending order.

Batch ID	Total Gold (mg)	Avg Gold (mg)	Total Silver (mg)	Avg Silver (mg)	Total Copper (mg)	Avg Copper (mg)
B001	125.4	25.1	98.7	19.7	76.2	15.2
B002	110.8	22.2	120.3	24.1	89.5	17.9
B003	95.6	19.1	87.4	17.5	102.8	20.6
B004	88.2	17.6	92.1	18.4	110.3	22.1
B005	80.5	16.1	85.9	17.2	95.7	19.1