**Aufgabe 1:**

1. **Formulieren Sie INSERT Statements, um die vorgegebene Datenbank mit folgenden Daten**

**zu erweitern.**

INSERT INTO Materialien (Herstellername)

VALUES

(Kupfer)

INSERT INTO Erhaltung (Bezeichnung)

VALUES

(Gut erhalten)

INSERT INTO Erhaltung (Bezeichnung)

VALUES

(Schön)

INSERT INTO Muenze (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser)

VALUES

(Lunar 1/20 Unze, 1.56, 1.56, 999.9, 14.6)

INSERT INTO Muenze (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser)

VALUES

(Lunar 5 Unzen, 155.518, 155.518, 999, 60.6)

INSERT INTO Muenze (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser)

VALUES

(Lunar 10 Unzen, 311.035, 311.035, 999, 60.6)

INSERT INTO Muenzen (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser, MaterialID)

VALUES

(1 Unze Maple Leaf, 31.104, 31.104, 999, 38, 2)

INSERT INTO Muenzen (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser, MaterialID)

VALUES

(1 Unze Mexico Libertad, 31.104, 31.104, 999, 40, 2)

INSERT INTO Muenzen (Bezeichnung, Rohgewicht, Feingewicht, Feinheit, Durchmesser, MaterialID)

VALUES

(Philharmoniker 1 Unze, 31.104, 31.104, 999.5, 37, 4)

1. **Was bedeutet referentielle Integrität und wieso ist es wichtig, dass man sie in Aufgabe 1a**

**beachtet?**

Referentielle Integrität bedeutet, dass jeder Fremdschlüssel einen korrespondierenden Primärschlüssel in einer anderen Tabelle haben muss. Das ist wichtig, um Dateninkonsistenz oder Datenanomalien zu verhindern.

In der 1 a ist die referentielle Integrität wichtig, um sicherzustellen, dass beim hinzufügen neuer Münzen die Münzen tatsächlich auf richtige Erhaltungszustände verweisen.

**Aufgabe 2:**

SELECT Rohgewicht, Feingewicht

FROM Muenzen

WHERE Gewicht < 30;



SELECT Bezeichnung, Durchmesser

FROM Muenzen

WHERE Bezeichnung LIKE Unze

SELECT \*

FROM Hersteller

WHERE Land IN ('GBR', 'MEX', 'CAN')

1. S

Aufgabe 4:

1. SELECT M.Bezeichnung, SUM(B.Menge) AS Gesamtmenge

FROM Muenzen M

JOIN Bestaende B ON M.MuenzeID = B.MuenzeID

GROUP BY M.Bezeichnung

ORDER BY Gesamtmenge DESC;

1. SELECT Serie, AVG(Feinheit) AS DurchschnittlicheFeinheit

FROM Muenzen

GROUP BY Serie

HAVING AVG(Feinheit) > 998;

Aufgabe 5:

a.

DELETE FROM Muenzen

WHERE Serie = 'Krügerrand';

b.

UPDATE Hersteller

SET Name = 'The Best Mint'

WHERE Name = 'Rand Refinery';

Aufgabe 6:

1. SELECT M.Bezeichnung, M.Erscheinungsjahr

FROM Muenzen M

WHERE M.Erscheinungsjahr > (

SELECT AVG(Erscheinungsjahr)

FROM Muenzen

);

1. SELECT S.Serie, M.Bezeichnung

FROM Serien S

LEFT JOIN Muenzen M ON S.Serie = M.Serie

ORDER BY S.Serie;