HW 3

Database Programming

DEVELOPING STORED PROCEDURES

Professor HG Locklear
hlocklear@pace.edu

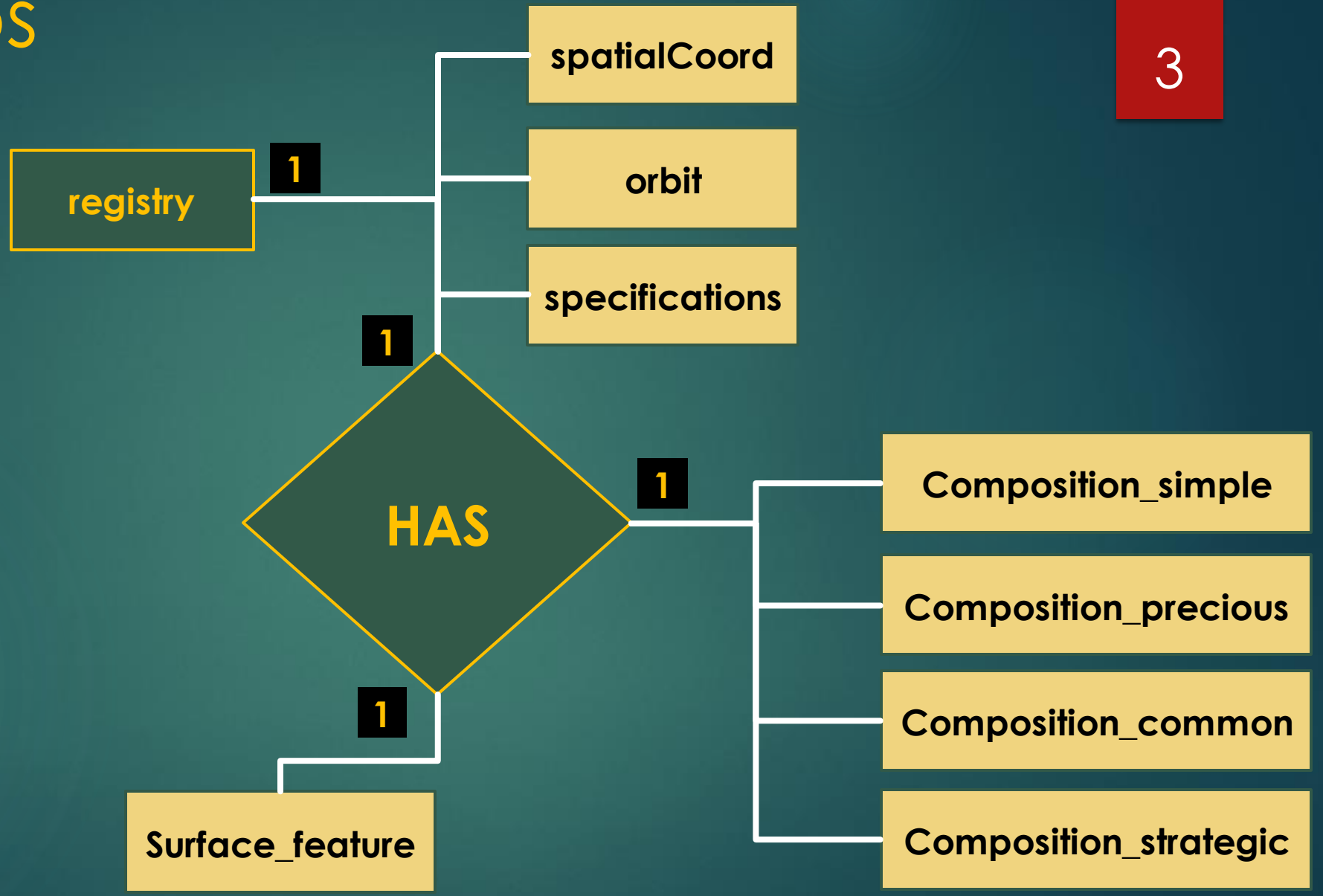
Asteroids

- **C-Type** (Carbonaceous) asteroids are the most common variety, forming around 75% of known asteroids. They are volatile-rich and distinguished by a very low albedo because their composition includes a large amount of carbon, in addition to rocks and minerals. They occur most frequently at the outer edge of the asteroid belt, 3.5 AU from the Sun, where 80% of the asteroids are of this type, whereas only 40% of asteroids at 2 AU from the Sun are C-type.

- **S-Type** (Siliceous) asteroids are asteroids with a spectral type that is indicative of a siliceous (i.e., stony) mineralogical composition. They are dominant in the inner part of the asteroid belt within 2.2 AU, common in the central belt within about 3 AU, but become rare farther out.

- **M-Type** (Metallic) asteroids are a spectral class of asteroids which appear to contain higher concentrations of metal phases (e.g., iron-nickel) than other asteroid classes, and are widely thought to be the source of iron meteorites.

Relationships



Relations

registry			
Designation	AType	Country	DDate

spatialCoord			
Designation	X	Y	Z

specifications					
Designation	Diameter	Mass	Density	Inclination	Rotation

surface_feature		
Designation	Surface	Water

orbit					
Designation	Aphelion	Perihelion	Eccentricity	Period_Orbit	Radius_Orbit

composition_simple		
Designation	Content_Rock	Content_Metal

composition_common				
Designation	Nickel	Molybdenum	Iron	Zinc

composition_precious							
Designation	Gold	Silver	Platinum	Palladium	Rhodium	Ruthenium	Iridium	Osmium

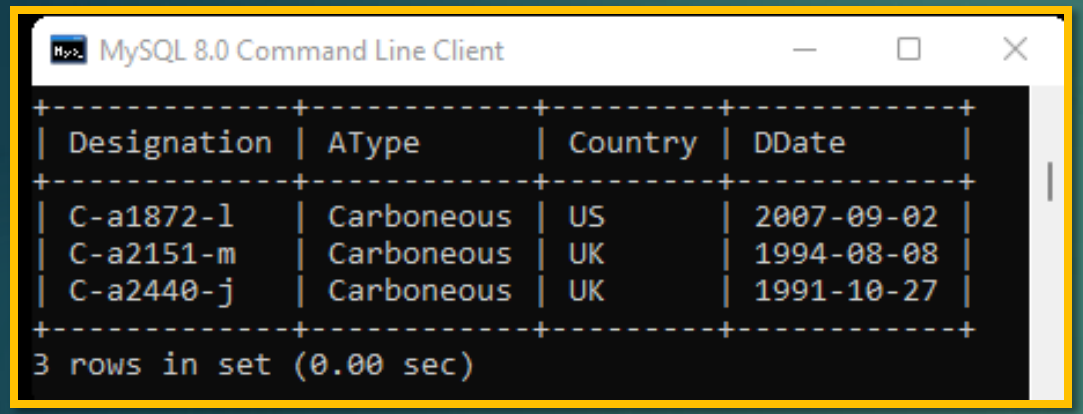
composition_strategic				
Designation	Chromium	Cobalt	Tungsten	Uranium

Units of Measure

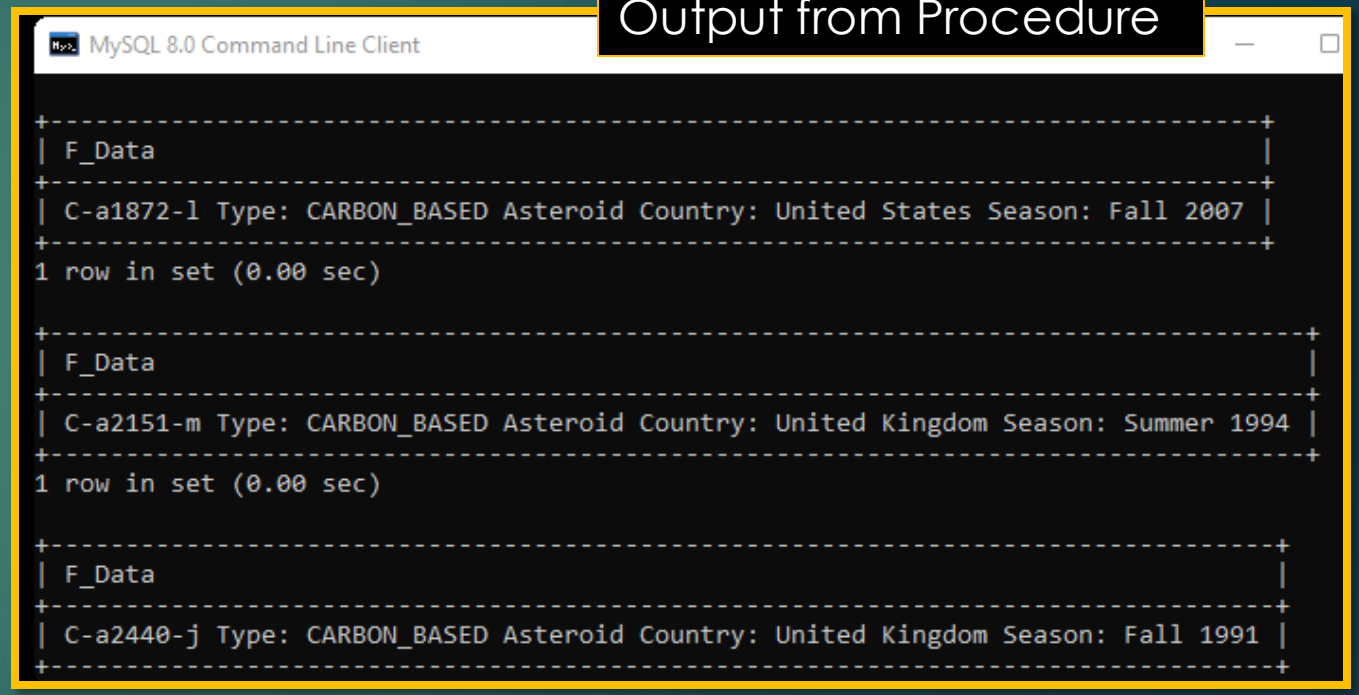
Attribute	Unit of Measure
Diameter	Meters
Mass	Kilograms
Density	Kilograms per Cubic Meter
Inclination	Degrees
Rotation	Hours
Aphelion	Astronomical Units
Perihelion	Astronomical Units
Eccentricity	Ratio
Period_Orbit	Years
Radius Orbit	Astronomical Units
X,Y, and Z	Number (Ordinate)
All Composition Attributes	Percentages of Mass
Water and Rock	Percentages

Stored Procedure 1

Create the Stored Procedure **showType** which accepts an Asteroid Type (**T**) and an integer (**C**)as its parameters and returns the data about the specified number (**C**) of asteroids of that type (**T**) in the format shown below.



Output from Procedure



US = **'United States'**
UK = **United Kingdom'**
RUSSIA = **'Russian Federation'**
CHINA = **People's Republic of China**

Carboneous = **'CARBON_BASED'**
Metallic = **'METAL_BASED'**
Silicaceous = **'SILICON_BASED'**

November – March = **'Winter'**
April – May = **'Spring'**
June – August = **'Summer'**
September – October = **'Fall'**

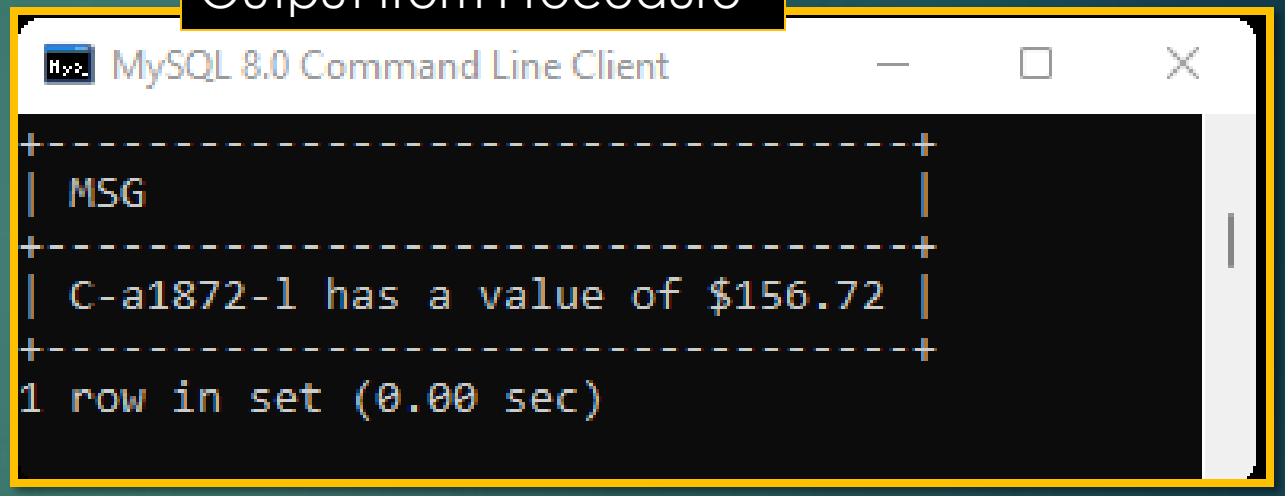
Stored Procedure 2

Create the Stored Procedure **showValue** which accepts an Asteroid Designation (**A**) and calculates the total value of its **strategic metals** and displays the total value in the format shown below.

Chromium = $12.50 per kg
Cobalt = $9.25 per kg
Tungsten = $7.75 per kg
Uranium = $10.00 per kg

The Total Value of an asteroids strategic metals is the sum of the values of each of the metals based on their percentage of the mass of the asteroid.

Output from Procedure



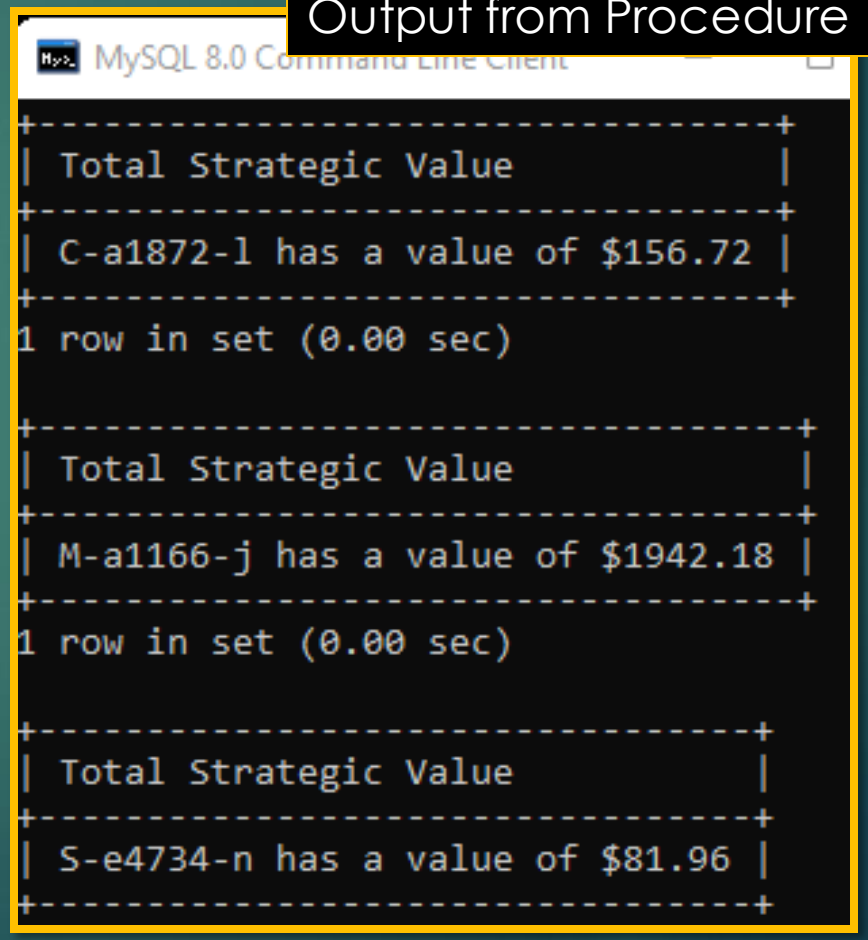
Stored Procedure 3

Create the Stored Procedure **showEachValue** which accepts a JSON Array (***of any length***) of Asteroid Designations (**A**) and calculates the total value of each of their strategic metals and displays the total values in the format shown below.

Chromium = $12.50 per kg
Cobalt = $9.25 per kg
Tungsten = $7.75 per kg
Uranium = $10.00 per kg

The Total Value of an asteroids strategic metals is the sum of the values of each of the metals based on their percentage of the mass of the asteroid.

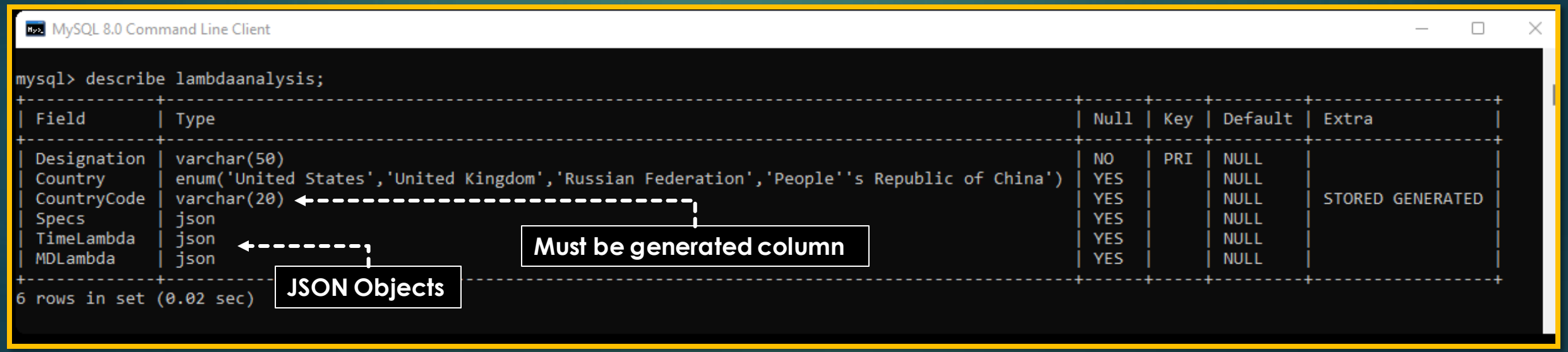
Output from Procedure



Stored Procedure 4

Create the Stored Procedure **specLambda** which accepts a JSON array ***(of any length)*** of asteroid designations and creates the table **lambdaAnalysis** as defined below based on the analysis procedures listed on the next slide.

The **lambdaAnalysis** table **must maintain** referential integrity with the **registry** table.

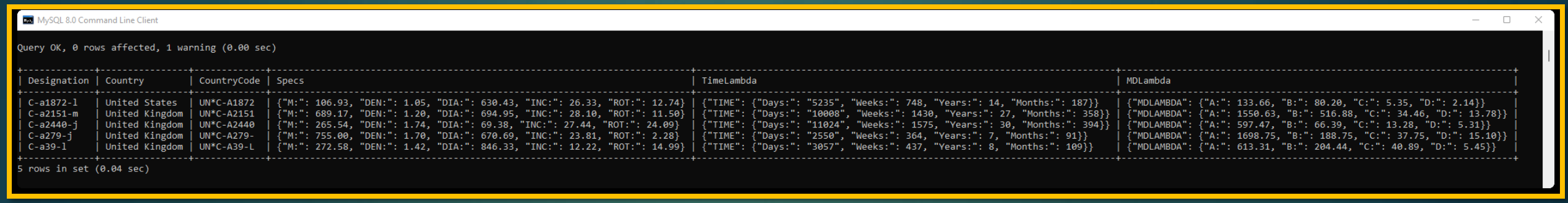


ALL DATA required to create the **lambdaAnalysis** table **must be** retrieved using functions and/or one or more cursors which utilize a loop.

Asteroid Analysis

Item	Protocol
Country	**US = 'United States' UK = 'United Kingdom RUS = 'Russian Federation CH = People's Republic of China**
CountryCode	First two characters of Country name concatenated with an '*' and the first 7 characters of the asteroid designation
Specs	'M' = Mass 'DEN' = Density 'DIA' = Diameter 'INC' = Inclination 'ROT' = Rotation
TimeLambda	Time between the DDate of the asteroid and Jan 1, 2022, in total days, total weeks, total months, and total years
MDLambda	A = Diameter Lambda if the diameter of the asteroid is more than four times greater than the asteroid's mass then then A = 125% of the asteroids mass. Otherwise, it is 225% of the asteroid's mass
	B = Density Lambda if the density of the asteroid is greater than 1.5 then B = 25% of the asteroids mass. Otherwise, it is 75% of the asteroid's mass
	C = Inclination Lambda if the inclination of the asteroid is greater than 15 then C = 5% of the asteroids mass. Otherwise, it is 15% of the asteroid's mass
	D = Rotation Lambda if the rotation of the asteroid is greater than 48 then D = 1% of the asteroids mass. Otherwise, it is 2% of the asteroid's mass

Task 4: Sample Output



MySQL 8.0 Command Line Client

```
Query OK, 0 rows affected, 1 warning (0.00 sec)

+-------------+---------------+-------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+
| Designation | Country       | CountryCode | Specs                                                                       | TimeLambda                                                                  | MDLambda                                                                    |
+-------------+---------------+-------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+
| C-a1872-l   | United States | UN*C-A1872  | {"M:": 106.93, "DEN:": 1.05, "DIA:": 630.43, "INC:": 26.33, "ROT:": 12.74} | {"TIME": {"Days:": "5235", "Weeks:": 748, "Years:": 14, "Months:": 187}}    | {"MDLAMBDA": {"A:": 133.66, "B:": 80.20, "C:": 5.35, "D:": 2.14}}           |
| C-a2151-m   | United Kingdom | UN*C-A2151  | {"M:": 689.17, "DEN:": 1.20, "DIA:": 694.95, "INC:": 28.10, "ROT:": 11.50} | {"TIME": {"Days:": "10008", "Weeks:": 1430, "Years:": 27, "Months:": 358}} | {"MDLAMBDA": {"A:": 1550.63, "B:": 516.88, "C:": 34.46, "D:": 13.78}}       |
| C-a2440-j   | United Kingdom | UN*C-A2440  | {"M:": 265.54, "DEN:": 1.74, "DIA:": 69.38, "INC:": 27.44, "ROT:": 24.09}  | {"TIME": {"Days:": "11024", "Weeks:": 1575, "Years:": 30, "Months:": 394}} | {"MDLAMBDA": {"A:": 597.47, "B:": 66.39, "C:": 13.28, "D:": 5.31}}          |
| C-a279-j    | United Kingdom | UN*C-A279-  | {"M:": 755.00, "DEN:": 1.70, "DIA:": 670.69, "INC:": 23.81, "ROT:": 2.28}  | {"TIME": {"Days:": "2550", "Weeks:": 364, "Years:": 7, "Months:": 91}}     | {"MDLAMBDA": {"A:": 1698.75, "B:": 188.75, "C:": 37.75, "D:": 15.10}}       |
| C-a39-l     | United Kingdom | UN*C-A39-L  | {"M:": 272.58, "DEN:": 1.42, "DIA:": 846.33, "INC:": 12.22, "ROT:": 14.99} | {"TIME": {"Days:": "3057", "Weeks:": 437, "Years:": 8, "Months:": 109}}    | {"MDLAMBDA": {"A:": 613.31, "B:": 204.44, "C:": 40.89, "D:": 5.45}}         |
+-------------+---------------+-------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+-----------------------------------------------------------------------------+
5 rows in set (0.04 sec)
```

```
CALL specLambda(JSON_ARRAY('C-a1872-l','C-a2151-m','C-a2440-j','C-a279-j','C-a39-l'));
```