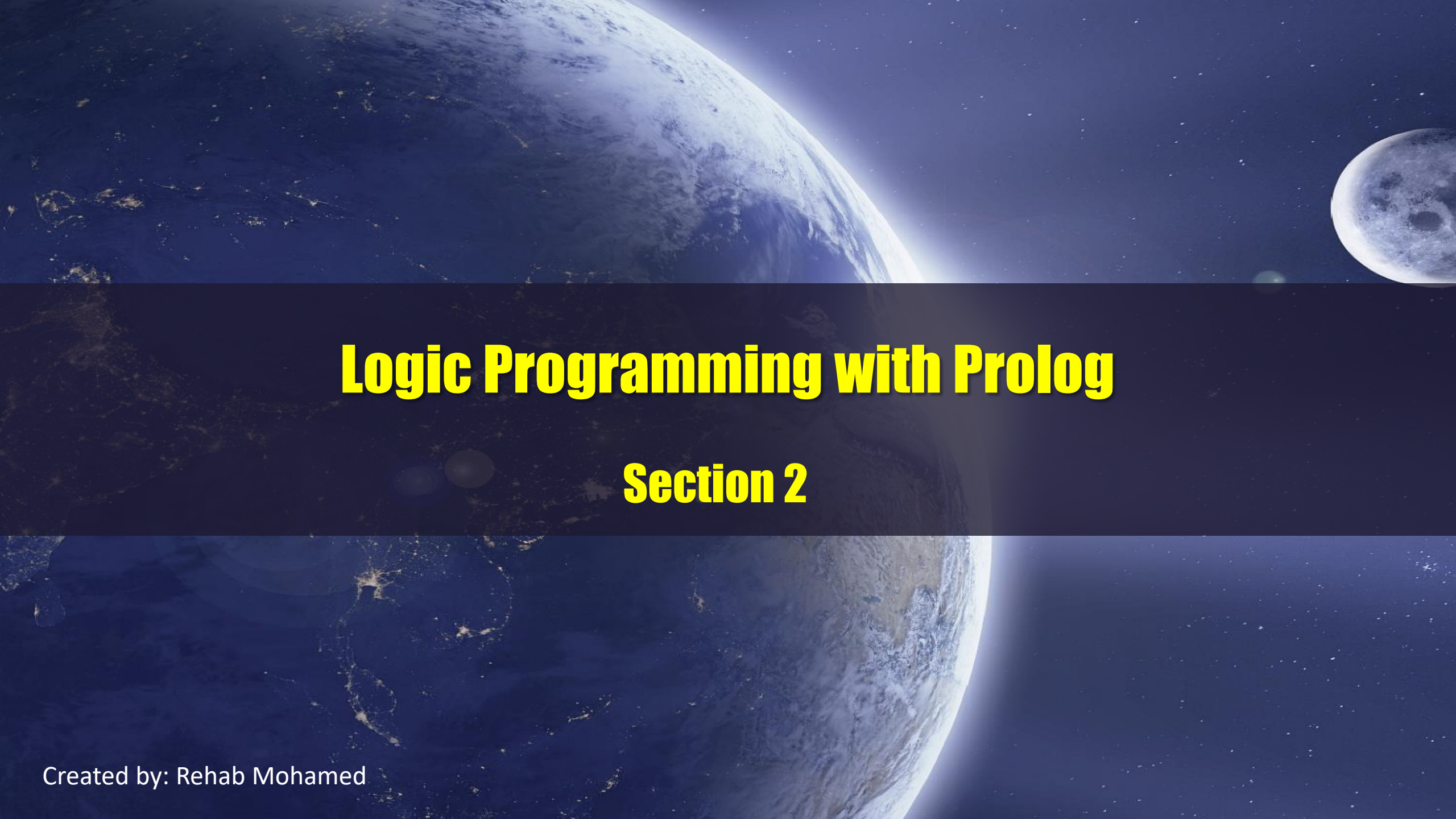


Logic Programming with Prolog

Section 2

Created by: Rehab Mohamed

What is prolog?

- Prolog stands for LOGical PROgramming (programming in logic). In the logic programming paradigm, prolog language is most widely available. Prolog is a declarative language, which means that a program consists of data based on the facts and rules (Logical relationship) rather than computing how to find a solution.
- Prolog is a declarative language that means we can specify what problem we want to solve rather than how to solve it.

Basics of prolog

- Used to solve problems involving
  - objects.
  - relationships between objects.

- Facts and rules use predicates which represent relationships among data objects.

Relationships

Example:

John owns the book

- The relationship: ownership
- The objects: book, John

Directional :

- John owns the book
- Not: The book owns John


Data Objects

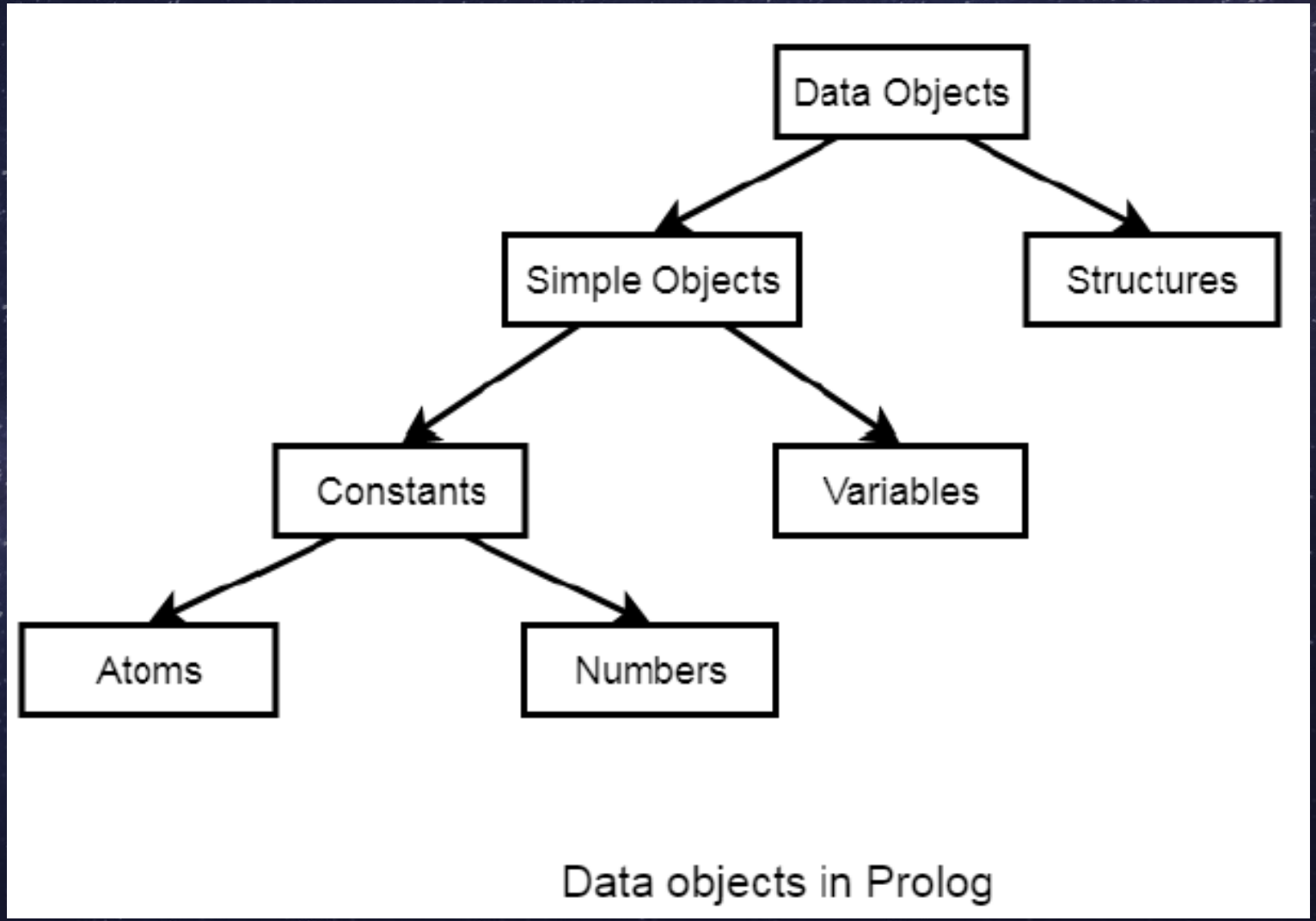
Data Objects

- The data objects in Prolog are called *terms*. Examples of terms that have been used in Prolog programs so far in this book are:

  mary , teacher(naama), X, teacher(X)

- There are several different types of term, which are listed below:
  - Atoms
  - Numbers
  - Variables
  - Structures
  - Predicates

Data Objects



Data objects in Prolog

Atoms

Atoms are constants that do not have numerical values. There are three ways in which atoms can be written:

(a) Any sequence of one or more letters (upper or lower case), numerals and underscores, beginning with a lower case letter, e.g.

john
today_is_Tuesday
fred_jones
a32_BCD

but not

Today
today-is-Tuesday
32abc

Atoms

(b) Any sequence of characters enclosed in single quotes, including spaces and upper case letters, e.g.

'Today is Tuesday'
'today-is-Tuesday'
'32abc'

(c) Any sequence of one or more special characters from a list that includes the following + - * / > < = & # @ :

Examples

+++
>=
>
+--

Numbers

All versions of Prolog allow the use of integers (whole numbers). They are written as any sequence of numerals from 0 to 9, optionally preceded by a + or - sign, for example:

623
-47
+5
025

Most versions of Prolog also allow the use of numbers with decimal points. They are written in the same way as integers, but contain a single decimal point, anywhere except before an optional + or - sign, e.g.

6.43
-.245
+256.

Variables

The name of a variable is denoted by any sequence of one or more letters (upper or lower case), numerals and underscores, beginning with an upper case letter or underscore, e.g.

X
Author
Person_A
_123A

but not

45_ABC
Person-A
author

Variables

Note: The variable _ which consists of just a single underscore is known as *the anonymous variable* and is reserved for a special purpose

Example:

```
male(ahmed).
male (mohamed).
```

```
?- male(X).
X = ahmed ;
X = mohamed.

?- male(_).
true .
```

Structures

- Objects that have many components.
- Begins with an atom, known here as a **functor**. The functor is followed by a sequence of one or more arguments, which are enclosed in brackets and separated by commas.
- The number of arguments a compound term has is called its **arity**. Some examples of compound terms are:

```
likes(paul,prolog)
person('john smith',32,doctor,london)
```

Structures

- Collection of Objects(atoms), Components, grouped together in one object
- Help Organize .
- Make code more readable.

- Example: Index Card for Library
  - Author's Name
  - Title
  - Date
  - Publisher
  - Name could be split also first, last, etc .

Structures

Example:

- `owns(john,book).`
- One Level:

```
owns(john,wuthering_heights).
owns(mary,moby_dick).
```

- Deeper:

```
owns(john,book(wuthering_heights,bronte)).
owns(john,book(wuthering_heights,
      author(emily,bronte))).
```

Fact:

owns(john, book(wuthering_heights, author(emily, bronte))).

Queries:

```
?- owns(john,X).
X = book(wuthering_heights, author(emily, bronte)).

?- owns(john,book(X,_)).
X = wuthering_heights.

?- owns(john,book(_,X)).
X = author(emily, bronte).

?- owns(john,book(_,_)).
true.

?- owns(john,book(_,author(X,_))).
X = emily.

?- owns(john,book(_,author(_,X))).
X = bronte.
```

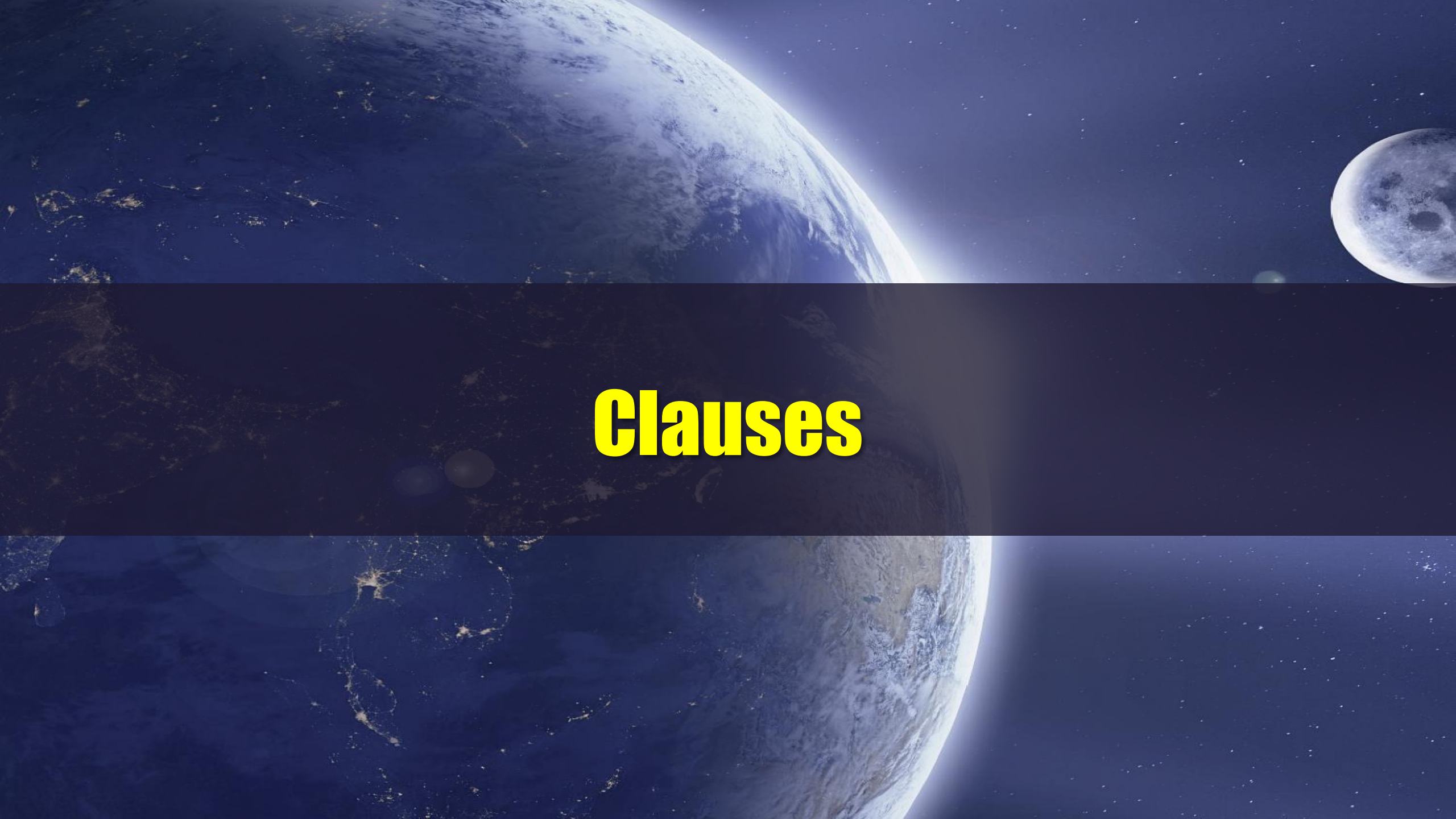
```
?- owns(john,book(X,author(Y,bronte))).
X = wuthering_heights,
Y = emily.
```

```
?- owns(john,book(_,author(_,bronte))).
true.
```

Predicate

- A predicate consists of a head and a number of arguments.
- Is a function which returns true/false.
- For example:

```
father(sam, pat).      %sam is father of pat
```


Clauses

Clauses

- Prolog program consists of a succession of *clauses*.
- There are two types of clause: *facts* and *rules*.

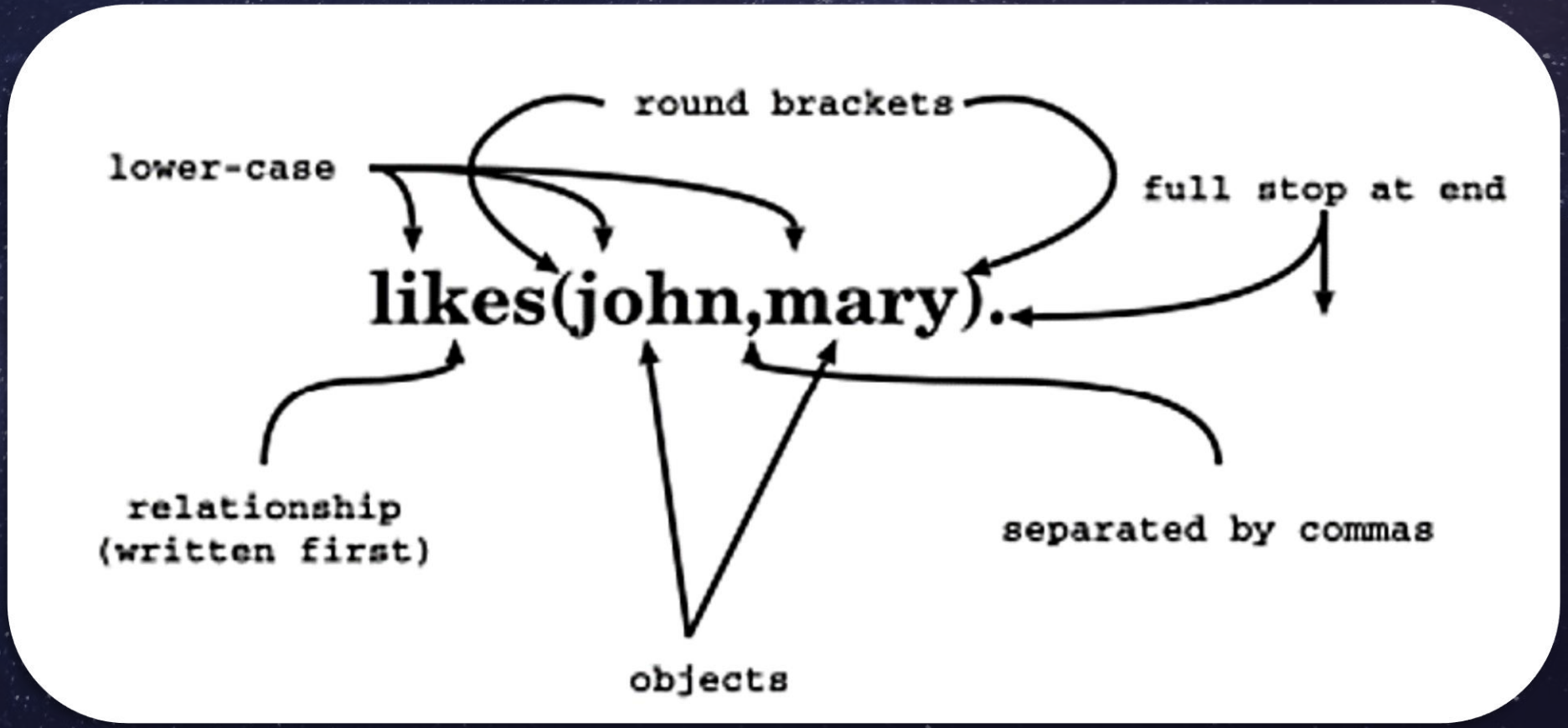
Facts

- Declare things that are unconditionally true.
- Have a head and empty body.
- Examples:

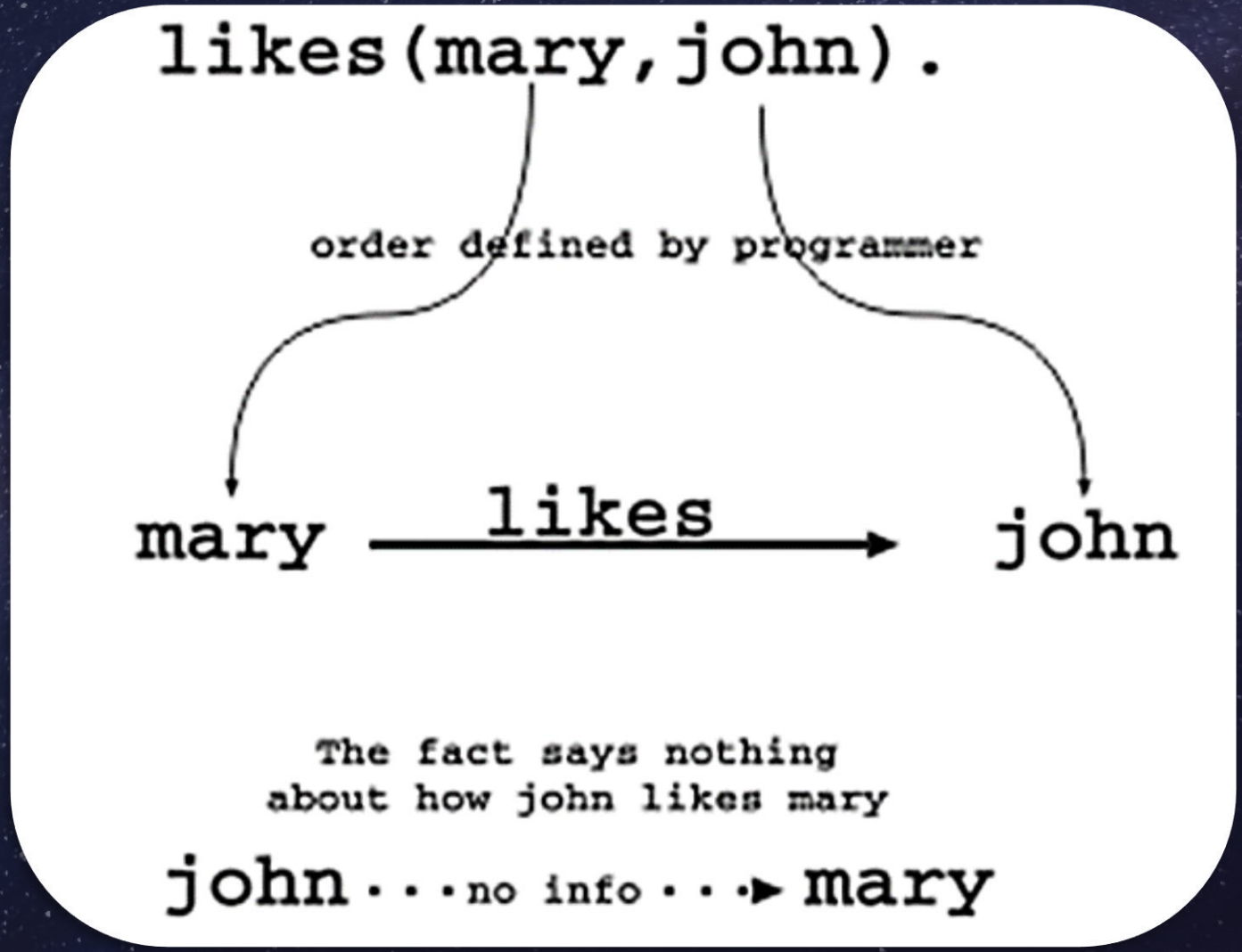
```
parent(tom, bob).      %tom is parent of bob
likes(joe, dogs).      % joe likes dogs
```

Facts

- Parts of facts:



Order of Objects



Examples of Facts

Examples:

Gold is valuable .
valuable(gold)

Jane is a female .
female(jane)

John owns some gold .
owns(john,gold)

John is the father of Mary .
father(john,mary)

Examples of Facts

Facts

```
likes(joe, fish).
likes(joe, mary).
likes(mary, book).
likes(john, book).
```

Query

```
?- likes(joe,mary).
true.

?- likes(mary,book).
true .

?- likes(fish,joe).
false.
```

Up until now questions just reflect exactly the database.

Does Mary like the book?

```
?- likes(mary,book).
```

More Interesting Question:
What objects does Mary like?

Variables.

```
?- likes(joe,X).
X = fish ;
X = mary.
```

```
?- likes(X,fish).
X = joe.
```

Rules

- Declare things (predicates) that are true depending on a given condition.
- Have a non-empty body.
- Example:

```
mother(X,Y):-parent(X,Y), female(X).
```

(X is mother of Y, if X is parent of Y and X is female)

Thank you