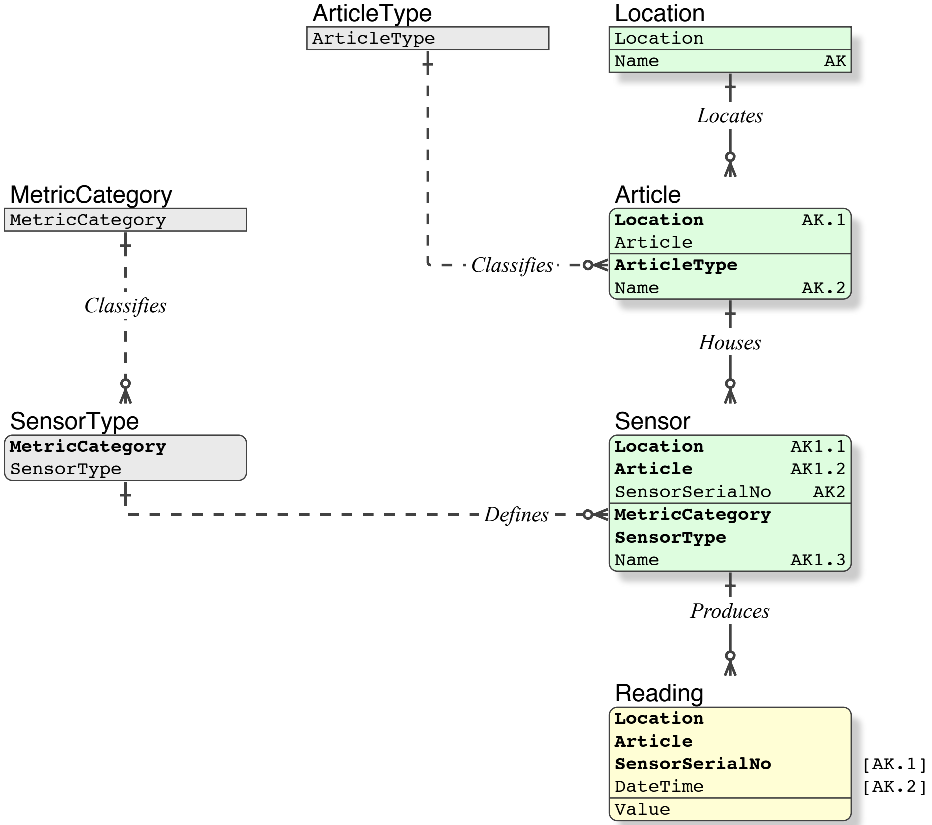
Sensor DM

Sensor

SOFTWARE GEMS



ArticleType
ArticleType

Location
Location
Name AK

Locates

MetricCategory
MetricCategory

Classifies

Article
Location AK.1
Article
ArticleType
Name AK.2

Classifies

Houses

SensorType
MetricCategory
SensorType

Defines

Sensor
Location AK1.1
Article AK1.2
SensorSerialNo AK2
MetricCategory
SensorType
Name AK1.3

Produces

Reading
Location
Article
SensorSerialNo [AK.1]
DateTime [AK.2]
Value

ArticleType
`Frame`
`CPU Board`
`Desktop`
`Laptop`
`Tablet`
`.or.`
`Fridge`
`Freezer`

MetricCategory
`Environment`
`System`

SensorType
`Environment.Temperature`
`Environment.Humidity`
`Environment.CO2`
`System.CPU`
`System.RAM`

Location
`Albany`
`Billings`
`Chicago`
`Detroit`
`.or.`
`Lounge`
`Dining`
`Bedroom2`

Note

- Re StackOverflow Question: **Relational Database schema design for metric storage**
- I have taken `MetricType` and `SensorType` to be synonymous.
- `Article` is shown as Dependent on (exists within) `Location`, alternately they could be separate vectors.
  In any case `Article` and `Location` together qualify `Sensor`.
- Since `SensorSerialNo` is unique (AK2), therefore `Reading(SensorSerialNo,DateTime)` is unique. An index is not required.
  However, in the event there are many queries on `Reading` via `SensorSerialNo` alone, such an index will boost performance.

Entity Type

⬜ Reference/Simple
🟩 Identifying
🟨 Transaction

IDEF1X Notation

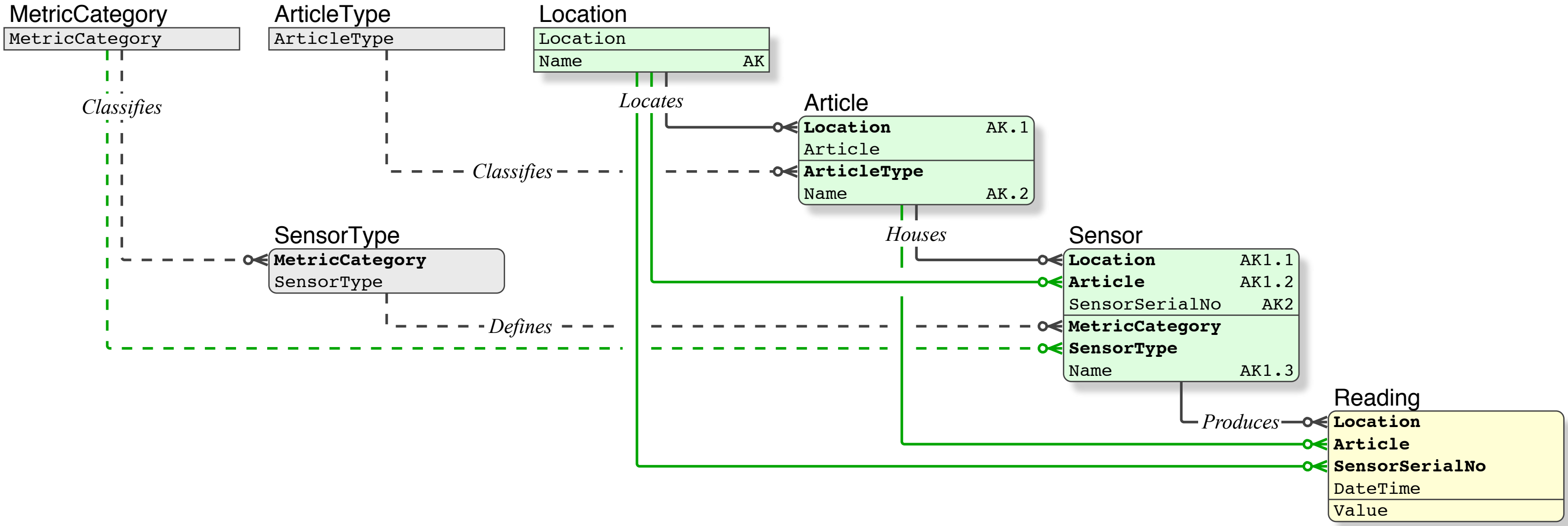
Sensor DM

JOIN

Those who are used to Record Filing Systems (characterised by Record IDs), the 1970's filth that is heavily marketed by "academics" as "relational", will not be aware of the Relational Power; Relational Integrity; or Relational Speed. Herein I will provide a *introductory* demonstration (not complete exposition) of:

- **Relational Power**, which is `JOIN` power, it massively reduces the number of `JOINs` required in all code segments: there is no need to `JOIN` the 'in-between' tables
- Relational Integrity (as distinct from Referential Integrity, which is physical only, and SQL feature), which is Logical, based on appropriate Relational Keys

The green relations are Logical Foreign Keys that result from the use of Relational Keys, these are to be 'read' and understood from the data model (previous page), only the black relations (physical) are required as `FOREIGN KEY REFERENCEs`. The "academics" do not understand SQL either.



MetricCategory

MetricCategory

ArticleType

ArticleType

Location

Location	
Name	AK

Article

Location	AK.1
Article	
ArticleType	
Name	AK.2

SensorType

MetricCategory
SensorType

Sensor

Location	AK1.1
Article	AK1.2
SensorSerialNo	AK2
MetricCategory	
SensorType	
Name	AK1.3

Reading

Location
Article
SensorSerialNo
DateTime
Value

Re Relational Power

```
-- List Location, Article; No of Sensors
SELECT
        L.Location,
        Name,       -- Location.Name
        Article,
        NumSensor = COUNT(SensorSerialNo)
    FROM     Sensor S,
        JOIN Location L ON (S.Location = L.Location)
    GROUP BY S.Location, Article
```

Re Relational Integrity

As a result of using Relational Keys, something a Record ID system cannot provide:

- An `Article` cannot be inserted or updated into an incorrect `Location`
- A `Sensor` cannot be inserted or updated into an incorrect `Article`
- A `Reading` cannot be inserted or updated into an incorrect `Sensor`