

# Census object

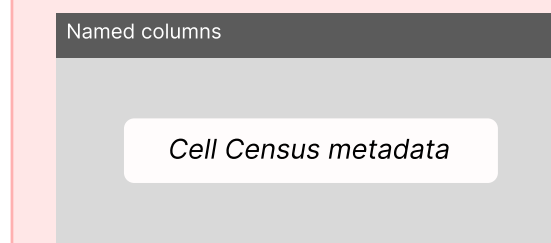
Collection

## "census\_info"

Collection

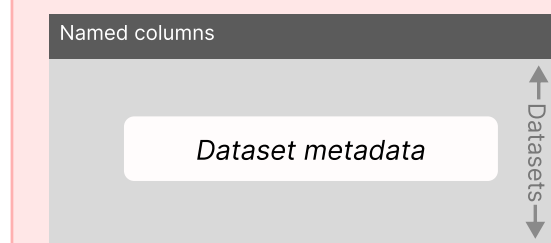
### "summary"

DataFrame



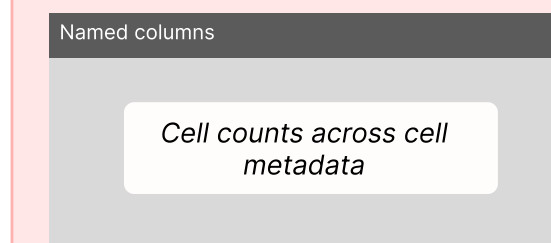
### "datasets"

DataFrame



### "summary\_cell\_counts"

DataFrame



## "census\_data"

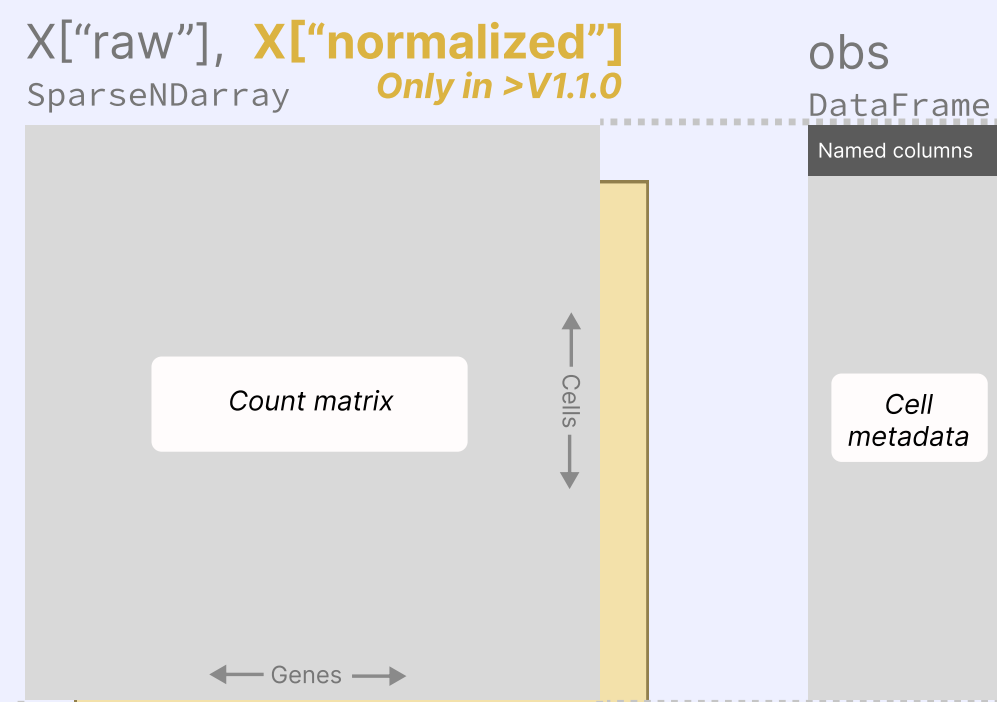
Collection

### "homo\_sapiens" or "mus\_musculus"

Experiment (is\_a Collection)

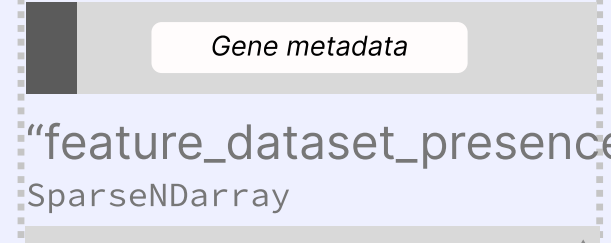
### ms["RNA"]

Measurement (is\_a Collection)



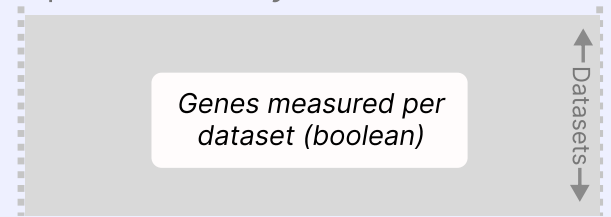
### var

DataFrame



### "feature\_dataset\_presence\_matrix"

SparseNDarray



## "census\_spatial" Contains Visium and Slide-seq data from CELLxGENE

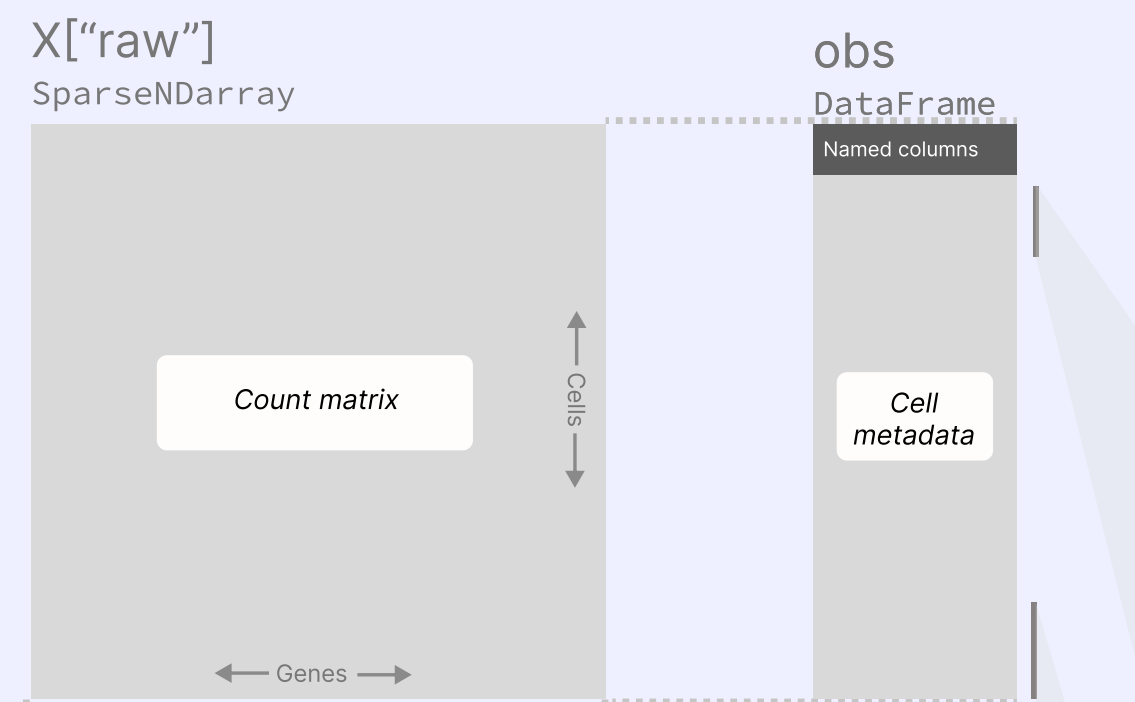
Collection

### "homo\_sapiens" or "mus\_musculus"

Experiment (is\_a Collection)

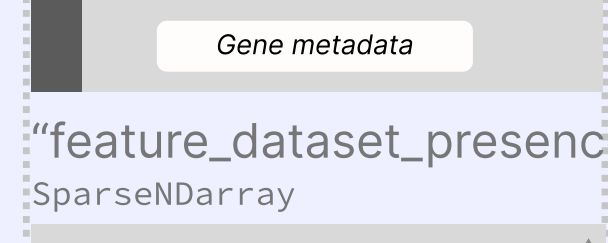
### ms["RNA"]

Measurement (is\_a Collection)



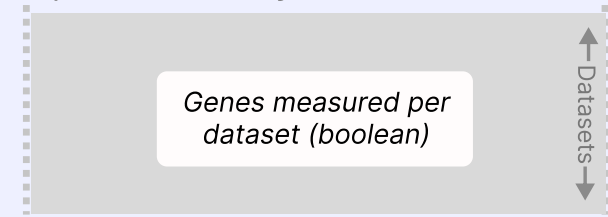
### var

DataFrame



### "feature\_dataset\_presence\_matrix"

SparseNDarray

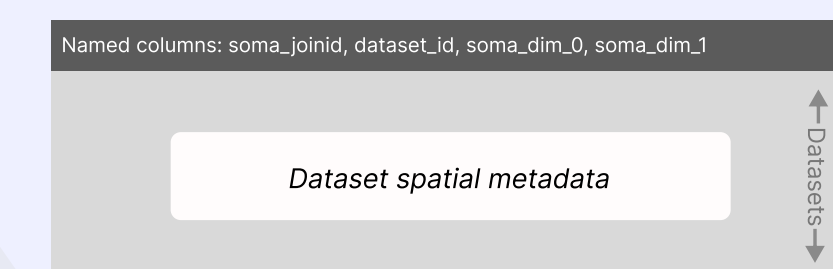


### spatial

Collection

### "scenes"

DataFrame



### "scene\_id\_A"

Scene (is\_a Collection)

"obs\_locations"  
DataFrame

exp  
Collection

"positions"  
SparseSpatialArray

"fullres\_image"  
DenseSpatialArray (only Visium)

"highres\_image"  
DenseSpatialArray (only Visium)

Extra obs column  
"scene\_id"

### "scene\_id\_N"

Scene (is\_a Collection)

"obs\_locations"  
DataFrame

exp  
Collection

"positions"  
SparseSpatialArray (only Visium)

"fullres\_image"  
DenseSpatialArray (only Visium)

"highres\_image"  
DenseSpatialArray (only Visium)

## spatial["scenes"]

DataFrame

Columns:

- soma\_joinid
- dataset\_id - corresponding to datasets.dataset\_id
- soma\_dim\_0 - name of the first dimension in coordinates
  - It must always be X
- soma\_dim\_1 - name of the Y column in coordinates
  - It must always be Y
- assay\_ontology\_term\_id - dup of obs.assay\_ontology\_term\_id
- assay - dup of obs.assay
- [IF VISIUM] tissue\_hires\_scalef -
- [IF VISIUM] fiducial\_diameter\_fullres -
- [IF VISIUM] spot\_diameter\_fullres

## spatial[scene\_id]["obs\_locations"]

DataFrame

Columns:

- soma\_joinid - corresponding to obs.soma\_joinid
- X - X coordinate, if Visium then pxl\_row\_in\_fullres
- Y - Y coordinate, if Visium then pxl\_col\_in\_fullres

## spatial[scene\_id].exp["positions"]

SparseSpatialArray

Columns:

- soma\_joinid - corresponding to obs.soma\_joinid
- X - X coordinate, if Visium then pxl\_row\_in\_fullres
- Y - Y coordinate, if Visium then pxl\_col\_in\_fullres
- array\_row - X array row number, if Visium then array\_row, else same as X.
- array\_col - Y array row number, if Visium then array\_col, else same as Y.
- soma\_geometry - radius of point, if Visium then spot\_diameter\_fullres/2, else it should be 0.003% of the radius occupied by the full cloud of points.

## spatial[scene\_id].exp["fullres\_image"]

## spatial[scene\_id].exp["highres\_image"]

DenseSpatialArray (only Visium)

OME TIFF

## Use cases

Slices of data using value filter into toolkits

```
human = census["census_spatial_data"]  
["homo_sapiens"]  
query = human.axis_query(  
    measurement_name = "RNA",  
    obs_query = tiledbsoma.AxisQuery(  
        value_filter = "cell_type == 'T cell'"  
    )  
)
```

query.to\_squidpy(masked=False) → list of SquidPy objects. Each item is a scene, either full data if masked is False, otherwise masked scenes only with relevant data. In the future masked can be augmented to neighborhood.

query.to\_seurat(masked=False) → list of Seurat objects. Same as above.

query.to\_spatialdata(masked=False) → list of SpatialData objects. Same as above. May change if there an encoding better than a list for SpatialData

query.X() → works as usual

query.obs() → works as usual

query.var() → works as usual

query.image(masked=False) → list of iters for image readers one per scene. Mask works as usual

query.geometry(masked=False) → list of iters for image readers one per scene. Mask works as usual

query.coords(masked=False) → list of iters for image readers one per scene. Mask works as usual