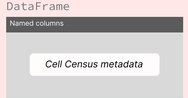
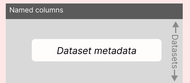
Census object
Collection

“census_info”
Collection

“summary”
DataFrame



“datasets”
DataFrame



“summary_cell_counts”
DataFrame

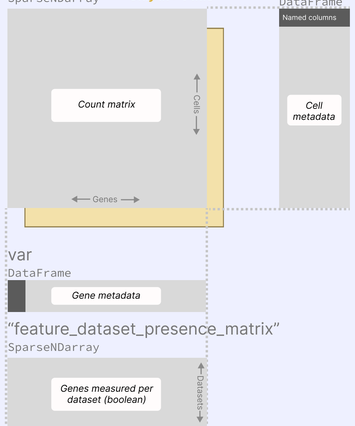
“census_data”
Collection

“homo_sapiens” or “mus_musculus”
Experiment (is_a Collection)

ms[“RNA”]
Measurement (is_a Collection)

X[“raw”], **X[“normalized”]** *Only in >V1.1.0*
SparseNDarray

obs
DataFrame



var
DataFrame

“feature_dataset_presence_matrix”
SparseNDarray

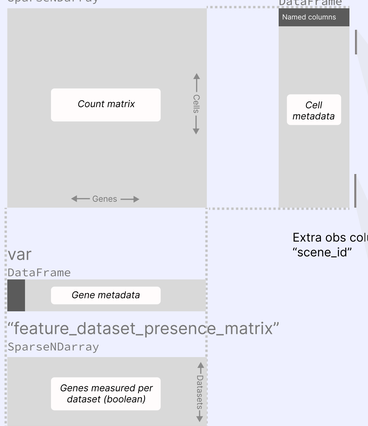
“census_spatial” Contains Visium and Slide-seq data from CELLxGENE
Collection

“homo_sapiens” or “mus_musculus”
Experiment (is_a Collection)

ms[“RNA”]
Measurement (is_a Collection)

X[“raw”]
SparseNDarray

obs
DataFrame



var
DataFrame

“feature_dataset_presence_matrix”
SparseNDarray

Extra obs column “scene_id”

spatial
Collection

“scenes”
DataFrame

“scene_id_A”
Scene (is_a Collection)

obsl
Collection

“positions”
SparseSpatialArray

“fullres_image”
DenseSpatialArray (only Visium)

“highres_image”
DenseSpatialArray (only Visium)

“scene_id_N”
Scene (is_a Collection)

obsl
Collection

“positions”
SparseSpatialArray

“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*].obsl[“positions”]
SOMAGeometryNDArray

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 Squid Py 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 that 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