

Census object
Collection

“census_info”
Collection

“summary”
DataFrame
Named columns
Cell Census metadata

“datasets”
DataFrame
Named columns
Dataset metadata

“summary_cell_counts”
DataFrame
Named columns
Cell counts across cell metadata

“census_data”
Collection

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

ms[“RNA”]
Measurement (is_a Collection)

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

obs
DataFrame
Named columns
Cell metadata

var
DataFrame
Gene metadata

“feature_dataset_presence_matrix”
SparseNDarray
Genes measured per dataset (boolean)

“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
Named columns
Cell metadata

var
DataFrame
Gene metadata

“feature_dataset_presence_matrix”
SparseNDarray
Genes measured per dataset (boolean)

spatial
Collection

“scenes”
DataFrame
Named columns: soma_joinid, dataset_id, soma_dim_0, soma_dim_1
Dataset spatial metadata

“scene_soma_joinid_A”
Scene (is_a Collection)
obs: Collection
“positions”: SparseSpatialArray
“fullres_image”: DenseSpatialArray (only Visium)
“highres_image”: DenseSpatialArray (only Visium)

“scene_soma_joinid_N”
Scene (is_a Collection)
obs: Collection
“positions”: SparseSpatialArray
“fullres_image”: DenseSpatialArray (only Visium)
“highres_image”: DenseSpatialArray (only Visium)

Extra obs column “scene_id”

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] spot_diameter_fullres

spatial[scene_id].obs[“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
- 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.
- 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.
- in_tissue - boolean indicating if spot is in tissue.

spatial[scene_id].exp[“fullres_image”]
spatial[scene_id].exp[“highres_image”]
SOMAImageNDArray (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
```