In [1]:
```
import scanpy as sc
import tiledb
import numpy as np
from sklearn.metrics import adjusted_mutual_info_score, adjusted_rand_score
```

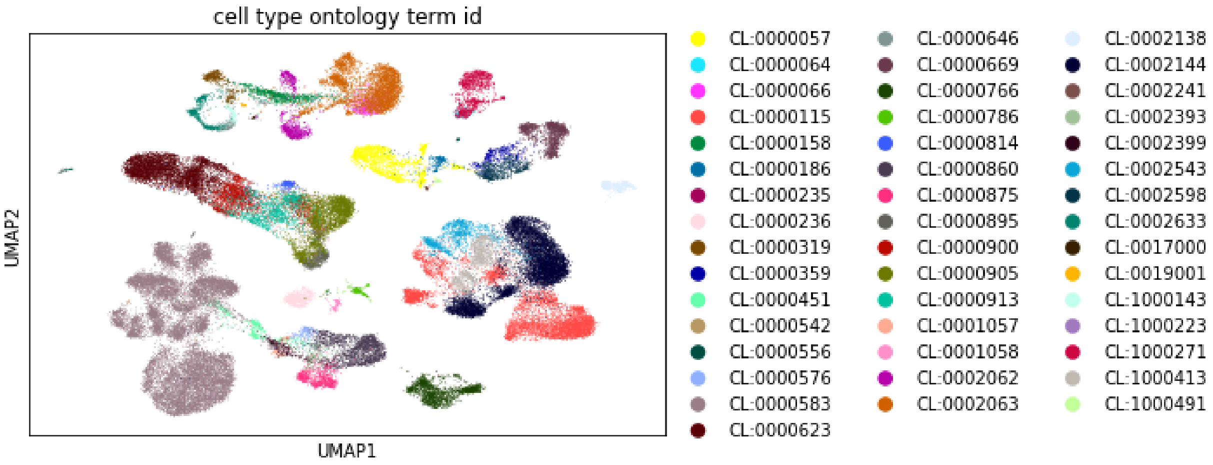
In [2]:
```
# load anndata (10x human lung cell atlas)
# https://cellxgene.cziscience.com/collections/5d445965-6f1a-4b68-ba3a-b8f765155d3a
adata = sc.read_h5ad('lung.h5ad')
```

In [3]:
```
# get marker genes from gene expression snapshot
X = tiledb.open('prod-cube/marker_genes/')
marker_genes_df = X.df[('UBERON:0002048','NCBITaxon:9606',[])]
marker_genes_df = marker_genes_df[marker_genes_df['effect_size_ttest'].notnull()]
```

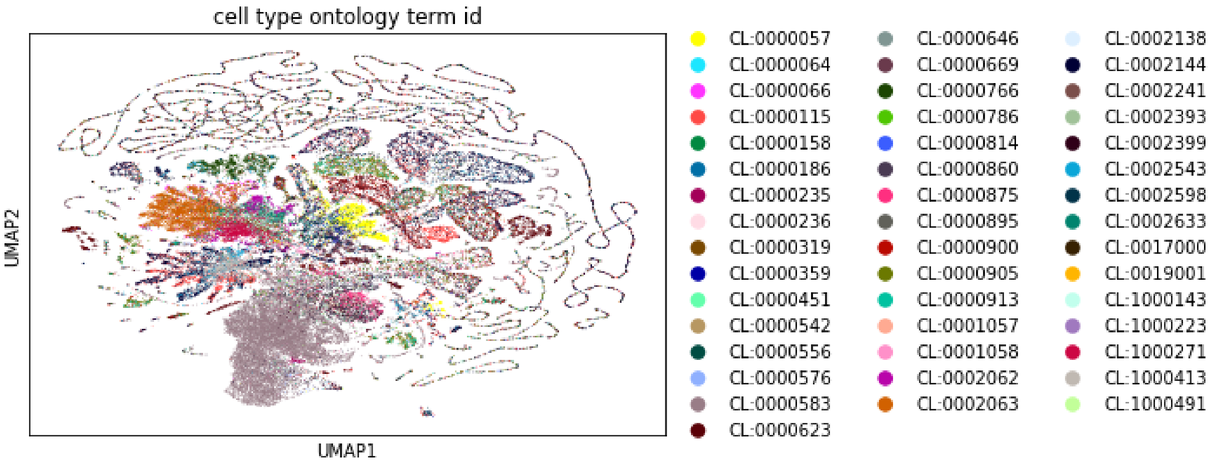
In [4]:
```
var_names = np.array(list(adata.var_names))
def agg_func(df):
    g = np.array(list(df['gene_ontology_term_id']))
    df = df[np.in1d(g,var_names)]
    x = df['effect_size_ttest']
    ix = np.argsort(x)[-5:]
    l = list(np.array(list(df['gene_ontology_term_id']))[ix])
    assert len(set(l)) == len(l)
    return l
marker_genes = list(set(np.concatenate(marker_genes_df.groupby('cell_type_ontology_term_
print('Found',len(marker_genes),'unique marker genes.')
```

```
Found 354 unique marker genes.
```

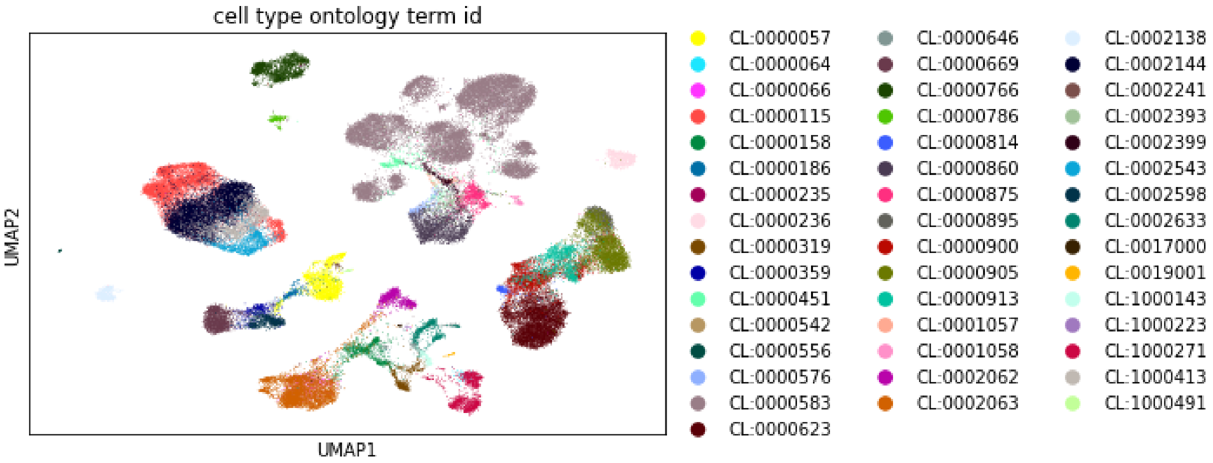
In [5]:
```
# analyze using standard workflow
sc.pp.highly_variable_genes(adata,n_top_genes=3000)
adata_orig = adata[:,adata.var['highly_variable']]
sc.tl.pca(adata_orig)
sc.pp.neighbors(adata_orig)
sc.tl.umap(adata_orig)
sc.pl.scatter(adata_orig,basis='umap',color='cell_type_ontology_term_id')
sc.tl.leiden(adata_orig)
```



In [6]:
```
# analyze using random genes (same number of genes as the selected markers)
random_genes = np.random.choice(adata.var_names,replace=False,size=len(marker_genes))
adata_rand = adata[:,random_genes]
sc.tl.pca(adata_rand)
sc.pp.neighbors(adata_rand)
sc.tl.umap(adata_rand)
sc.pl.scatter(adata_rand,basis='umap',color='cell_type_ontology_term_id')
sc.tl.leiden(adata_rand)
```



In [7]:
```
# analyze using marker genes
adata_sub = adata[:,marker_genes]
sc.tl.pca(adata_sub)
sc.pp.neighbors(adata_sub)
sc.tl.umap(adata_sub)
sc.pl.scatter(adata_sub,basis='umap',color='cell_type_ontology_term_id')
sc.tl.leiden(adata_sub)
```



In [8]:
```
ari_orig = adjusted_rand_score(adata.obs['cell_type_ontology_term_id'], adata_orig.obs['
ari_sub = adjusted_rand_score(adata.obs['cell_type_ontology_term_id'], adata_sub.obs['le
ari_rand = adjusted_rand_score(adata.obs['cell_type_ontology_term_id'], adata_rand.obs['
```

In [9]:
```
_orig = adjusted_mutual_info_score(adata.obs['cell_type_ontology_term_id'], adata_orig.ob
_sub = adjusted_mutual_info_score(adata.obs['cell_type_ontology_term_id'], adata_sub.obs[
_rand = adjusted_mutual_info_score(adata.obs['cell_type_ontology_term_id'], adata_rand.ob
```

In [10]:

```
print("Adjusted rand score")
print("Default",ari_orig)
print("Markers",ari_sub)
print("Random",ari_rand)
print("\nNormalized mutual information")
print("Default",nmi_orig)
print("Markers",nmi_sub)
print("Random",nmi_rand)
```

```
Adjusted rand score
Default 0.5039795188353338
Markers 0.5010254679525885
Random 0.1848075330662739

Normalized mutual information
Default 0.800698420253354
Markers 0.7915457877376938
Random 0.40771650978491886
```