Blob Matching Optimization

Summary of work and next steps

Samuel Alter | Reveal Global Consulting | 2025-03-28

Agenda

1. Introduction

Agenda

1. Introduction
2. Work Timeline

Agenda

1. Introduction
2. Work Timeline
3. Performance Results

Agenda

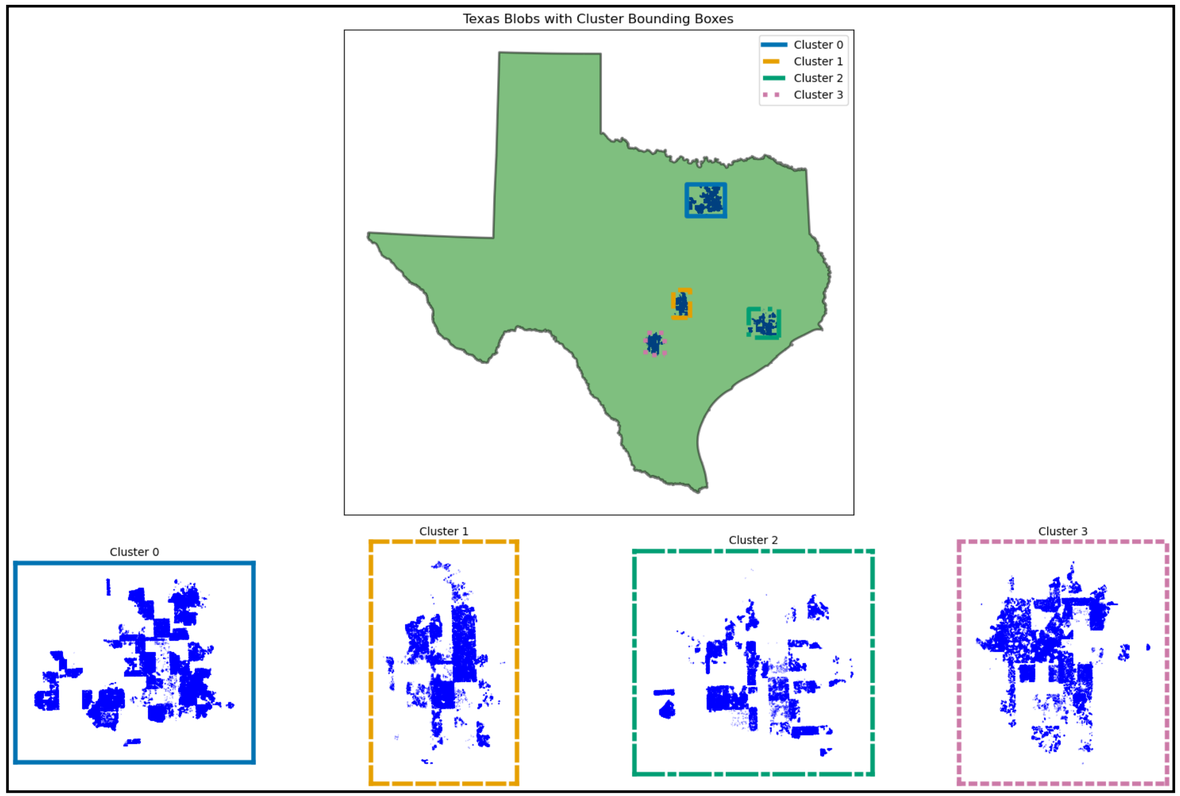
1. Introduction
2. Work Timeline
3. Performance Results
4. Next Steps

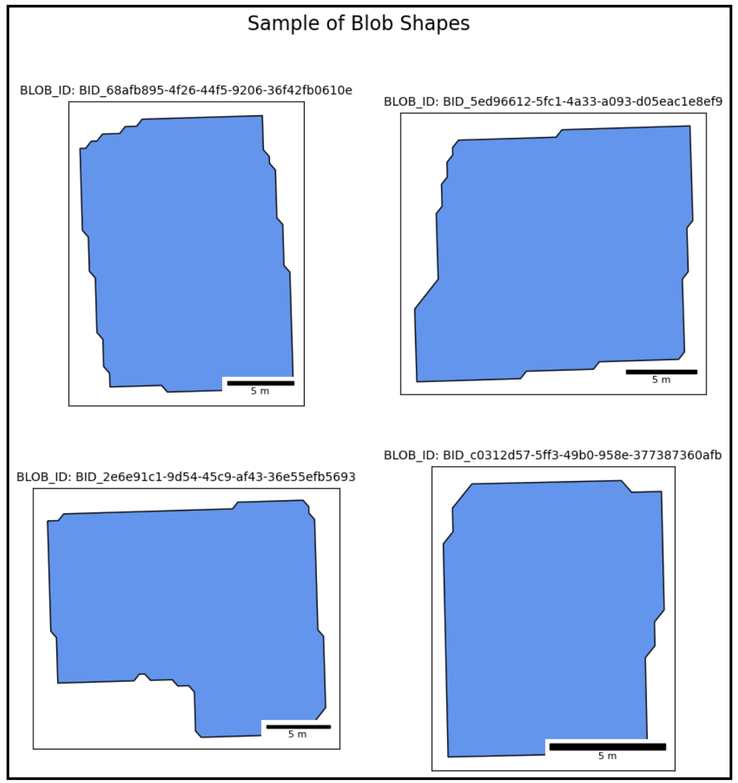
Agenda

1. **Introduction**
2. Work Timeline
3. Performance Results
4. Next Steps

Introduction

Blob Matching





Agenda

1. Introduction
2. **Work Timeline**
3. Performance Results
4. Next Steps

Work Timeline

Overview

1. Understand functions

Work Timeline

Overview

1. Understand functions
2. Benchmark process

Work Timeline

Overview

1. Understand functions
2. Benchmark process
3. Isolate inefficient areas

Work Timeline

Overview

1. Understand functions
2. Benchmark process
3. Isolate inefficient areas
4. Optimize the inefficient functions

Work Timeline

Overview

1. Understand functions
2. Benchmark process
3. Isolate inefficient areas
4. Optimize the inefficient functions
5. Benchmark optimized process

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script
4. Write benchmarking tools to measure and analyze performance of script

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script
4. Write benchmarking tools* to measure and analyze performance of script

   *Tools:

   - @Timer decorator
   - @ErrorCatcher decorator
   - Results analysis CLI script

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script
4. Write benchmarking tools to measure and analyze performance of script
5. Get performance results with limited, 835-row sample dataset and…

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script
4. Write benchmarking tools to measure and analyze performance of script
5. Get performance results with limited, 835-row sample dataset and…
6. …Get performance results with larger, ~6,000-row sample dataset

Work Timeline

Details

1. Isolate blob matching functions within `BlobSearchBusinessClass.py`
2. Bring in supporting functions and scripts
3. Refocus with `BlobMatchingBusinessClass.py` script
4. Write benchmarking tools to measure and analyze performance of script
5. Get performance results with limited, 835-row sample dataset and…
6. …Get performance results with larger, ~6,000-row sample dataset
7. Determine which functions within script are least efficient

Work Timeline

Details

1. Add logging messages throughout script

Work Timeline

Details

1. Add logging messages throughout script
2. Build benchmarking tools (@Timer/ErrorCatcher, and results analysis script)

Work Timeline

Details

1. Add logging messages throughout script
2. Build benchmarking tools (@Timer/ErrorCatcher, and results analysis script)
3. Add code benchmarks to functions

Work Timeline

Details

1. Add logging messages throughout script
2. Build benchmarking tools (@Timer/ErrorCatcher, and results analysis script)
3. Add code benchmarks to functions
4. Improvements to execution speed: swap ‘`spawn`’ for ‘`fork`’

Work Timeline

Details

1. Add logging messages throughout script
2. Build benchmarking tools (@Timer/ErrorCatcher, and results analysis script)
3. Add code benchmarks to functions
4. Improvements to execution speed: swap ‘`spawn`’ for ‘`fork`’

```
'fork'
```

```
mp.set_start_method('spawn', force=True)
```

Work Timeline

Details

1. Add logging messages throughout script
2. Build benchmarking tools (@Timer/ErrorCatcher, and results analysis script)
3. Add code benchmarks to functions
4. **Improvements to execution speed: swap ‘`spawn`’ for ‘`fork`’**

```
'fork'
```

```
mp.set_start_method('spawn', force=True)
```

Improvements in Execution Speed

Time Savings: ~40x

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_spawn.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5): 100%|##########| 1/1 [00:00<00:00, 444.26iteration/s]
Process GeoHash (1): 1iteration [03:10, 190.15s/iteration]

Before Impute Duplication Removal
NEW BLOB BC MATCHES     : 354
NEW BLOB BC IMPUTED     : 49
NEW BLOB IMPUTED        : 49
MARK INVALID BLOB BC    : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES     : 354
NEW BLOB BC IMPUTED     : 49
NEW BLOB IMPUTED        : 49
No Blob Business Classes to Mark as Invalid
Total Time: 196.3545618057251 seconds


Total Time: 196.35533475875854 seconds
```

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_fork.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5): 100%|##########| 1/1 [00:00<00:00, 9642.08iteration/s]
Process GeoHash (1): 1iteration [00:02,  2.62s/iteration]

Before Impute Duplication Removal
NEW BLOB BC MATCHES     : 354
NEW BLOB BC IMPUTED     : 49
NEW BLOB IMPUTED        : 49
MARK INVALID BLOB BC    : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES     : 354
NEW BLOB BC IMPUTED     : 49
NEW BLOB IMPUTED        : 49
No Blob Business Classes to Mark as Invalid
Total Time: 4.780044078826904 seconds


Total Time: 4.780802965164185 seconds
```

Improvements in Execution Speed

Time Savings: ~40x

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_spawn.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5): 100%|##########| 1/1 [00:00<00:00, 9642.26iteration/s]
Process GeoHash (1): 1iteration [03:16, 196.35s/iteration]

Before Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
MARK INVALID BLOB BC     : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
No Blob Business Classes to Mark as Invalid
Total Time: 196.3545618057251 seconds


Total Time: 196.35533475875854 seconds
```

196 seconds

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_fork.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5): 100%|##########| 1/1 [00:00<00:00, 9642.08iteration/s]
Process GeoHash (1): 1iteration [00:02,  2.62s/iteration]

Before Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
MARK INVALID BLOB BC     : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
No Blob Business Classes to Mark as Invalid
Total Time: 4.780044078826904 seconds


Total Time: 4.780802965164185 seconds
```

Improvements in Execution Speed

Time Savings: ~40x

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_spawn.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5)
Process GeoHash (1):

Before Impute Duplica
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
MARK INVALID BLOB BC     : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
No Blob Business Classes to Mark as Invalid
Total Time: 196.3545618057251 seconds


Total Time: 196.35533475875854 seconds
```

196 seconds

```
% python3.9 ./app/BlobSearch/BlobSearchBusinessClass_original_fork.py 202407 --geo_hash
"9vgm0d" --p 7
Args: Namespace(yyyymm='202407', city=None, county=None, state=None, full_sat_img_id=None,
geo_hash='9vgm0d', p=7, info=False, override_history_check=False, overwrite=False)
Num Geohashes: 1
BlobSearchBusinessClass PROCESS AREA: City None | County None | State None | FullSatImgID None
| GeoHash 9vgm0d
OVERRIDE_HISTORY_CHECK = False
OVERWRITE              = False
CURR_YYYYMM                          : 202407
PREV_YYYYMM                          : 202406
Number of Geo Hashes                 : 1
Number of Large Geo Hashes           : 1 | ['9vgm0d']
Number of Previous Blobs             : 400
Number of Current Blobs              : 435
Curr Blob BCs (with Invalid)         : 0
Curr Blob BCs (without Invalid)      : 0
Processing 5 Processes with Batch Size 1
Process GeoHashes (5): 100%
Process GeoHash (1): 1itera

Before Impute Duplication R
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
MARK INVALID BLOB BC     : 0

After Impute Duplication Removal
NEW BLOB BC MATCHES      : 354
NEW BLOB BC IMPUTED      : 49
NEW BLOB IMPUTED         : 49
No Blob Business Classes to Mark as Invalid
Total Time: 4.780044078826904 seconds


Total Time: 4.780802965164185 seconds
```

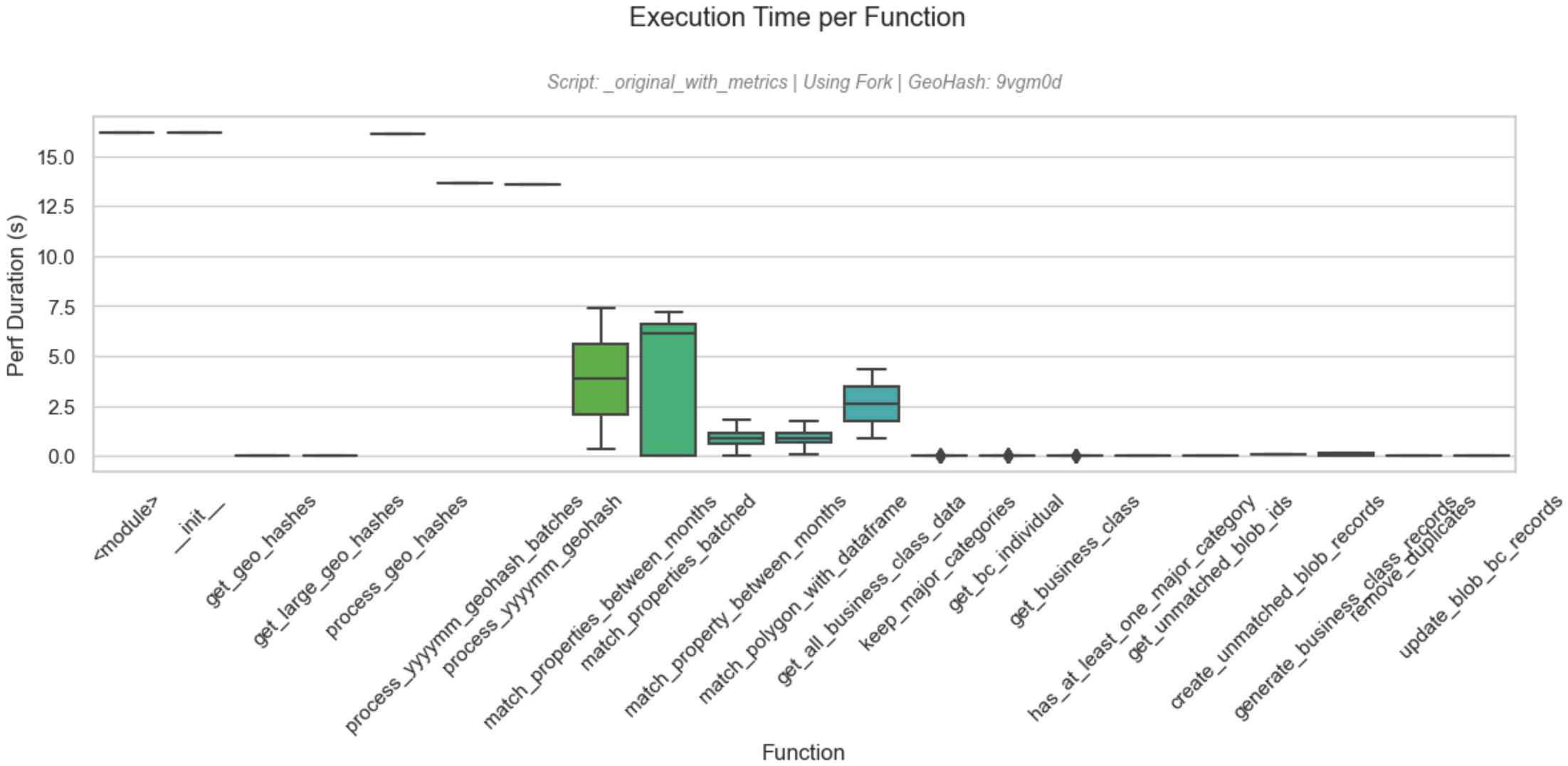
<5 seconds

Agenda

1. Introduction
2. Work Timeline
3. **Performance Results**
4. Next Steps

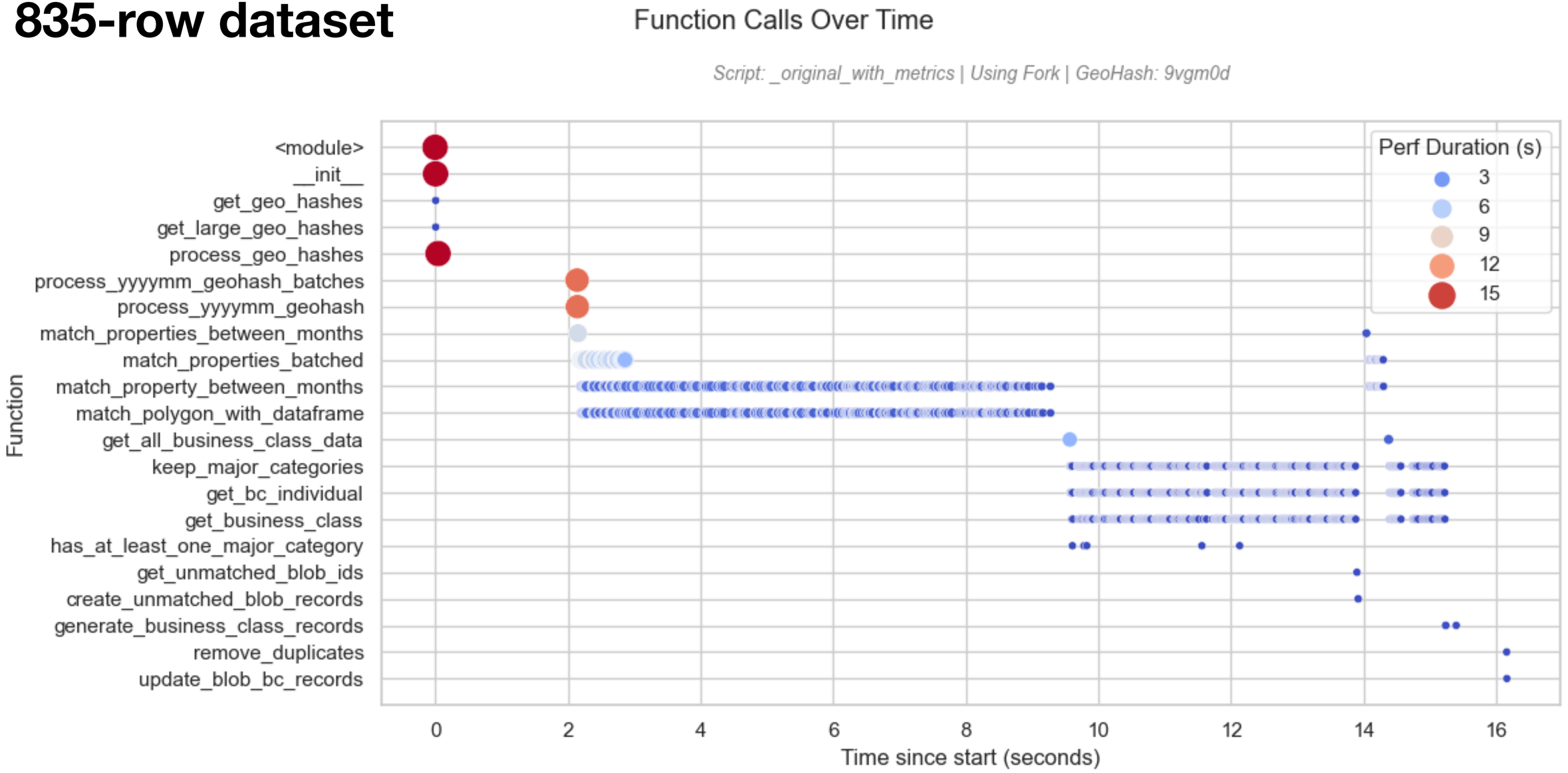
Performance Results

835-row dataset



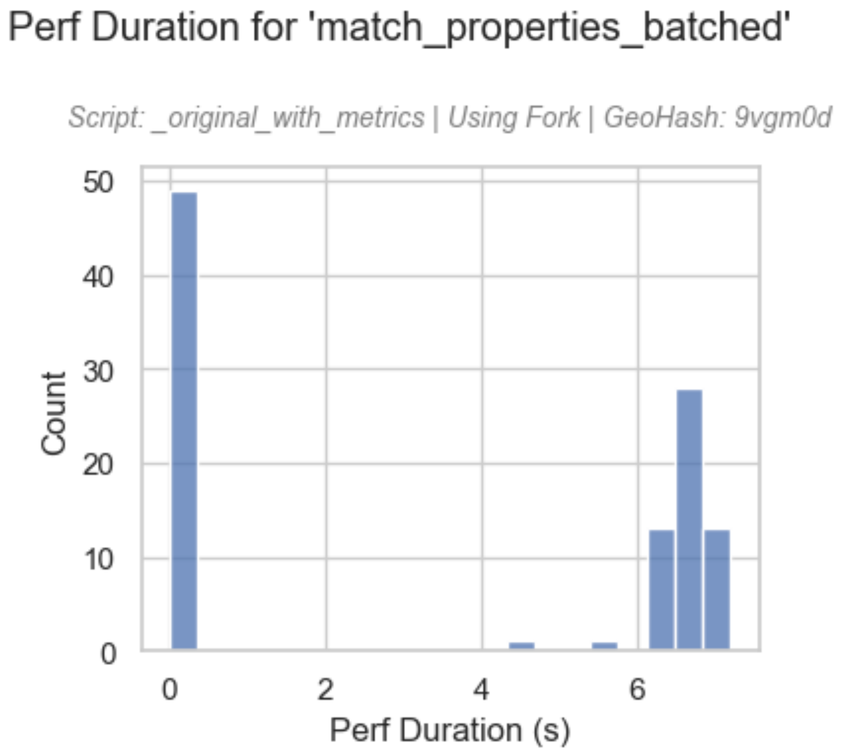
Performance Results

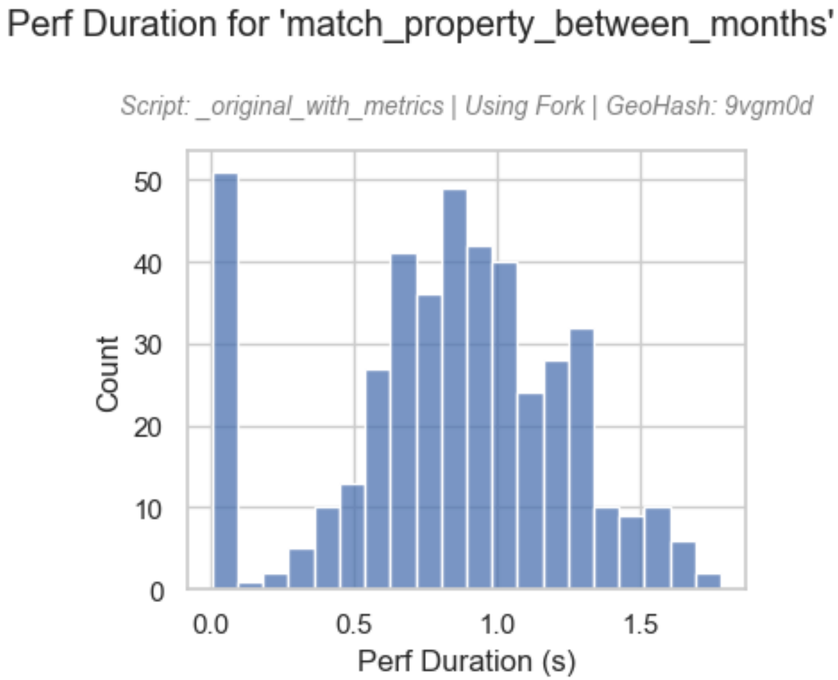
835-row dataset

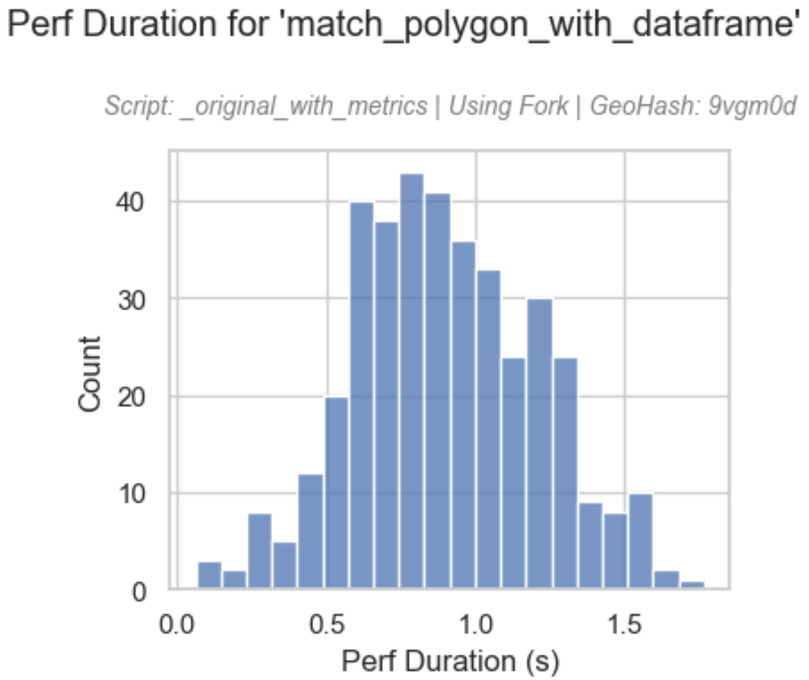


Performance Results

835-row dataset

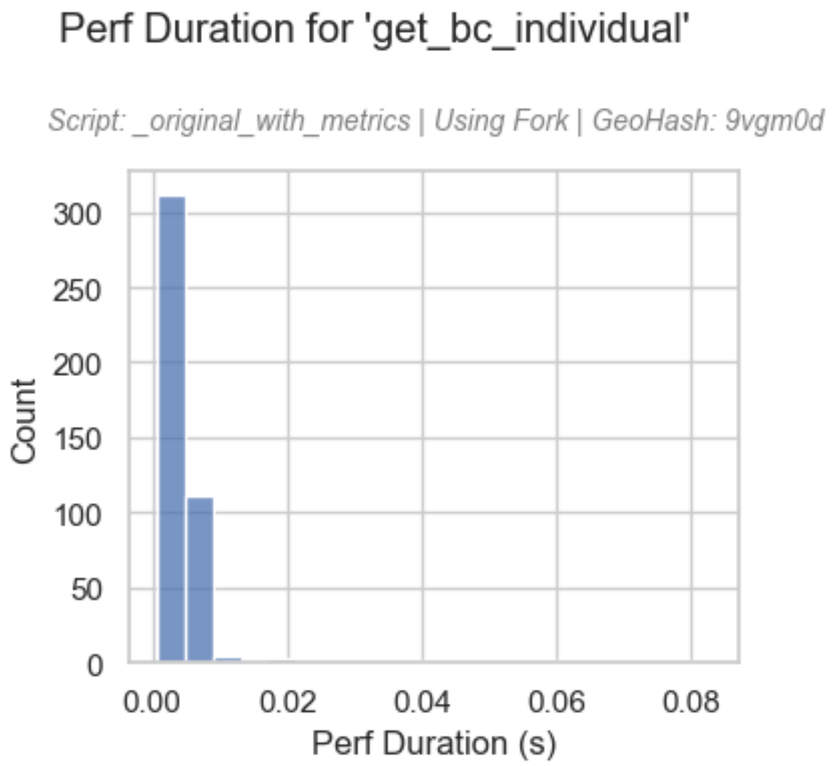


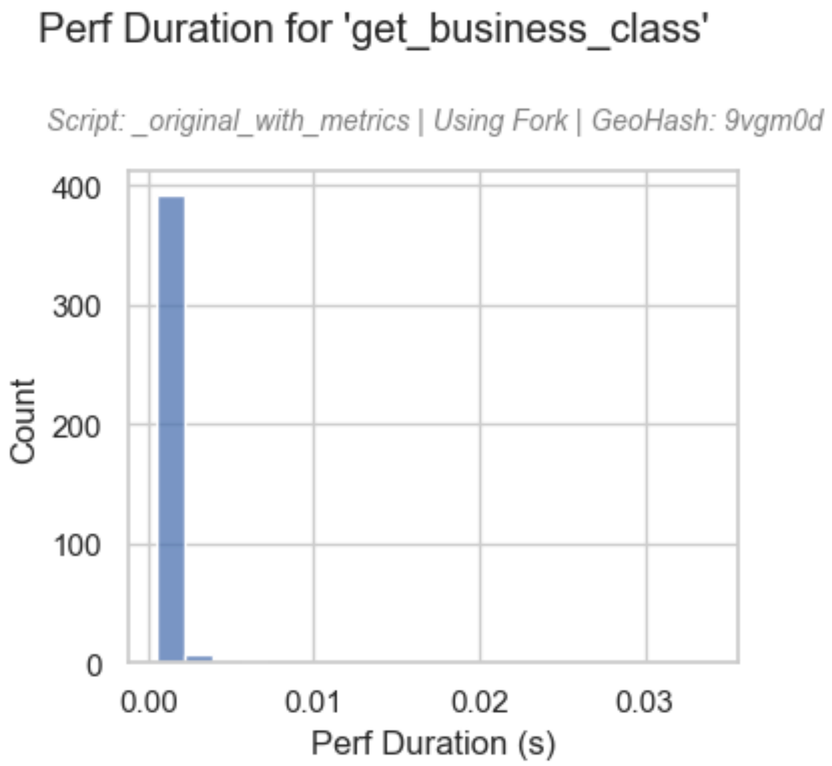




Performance Results

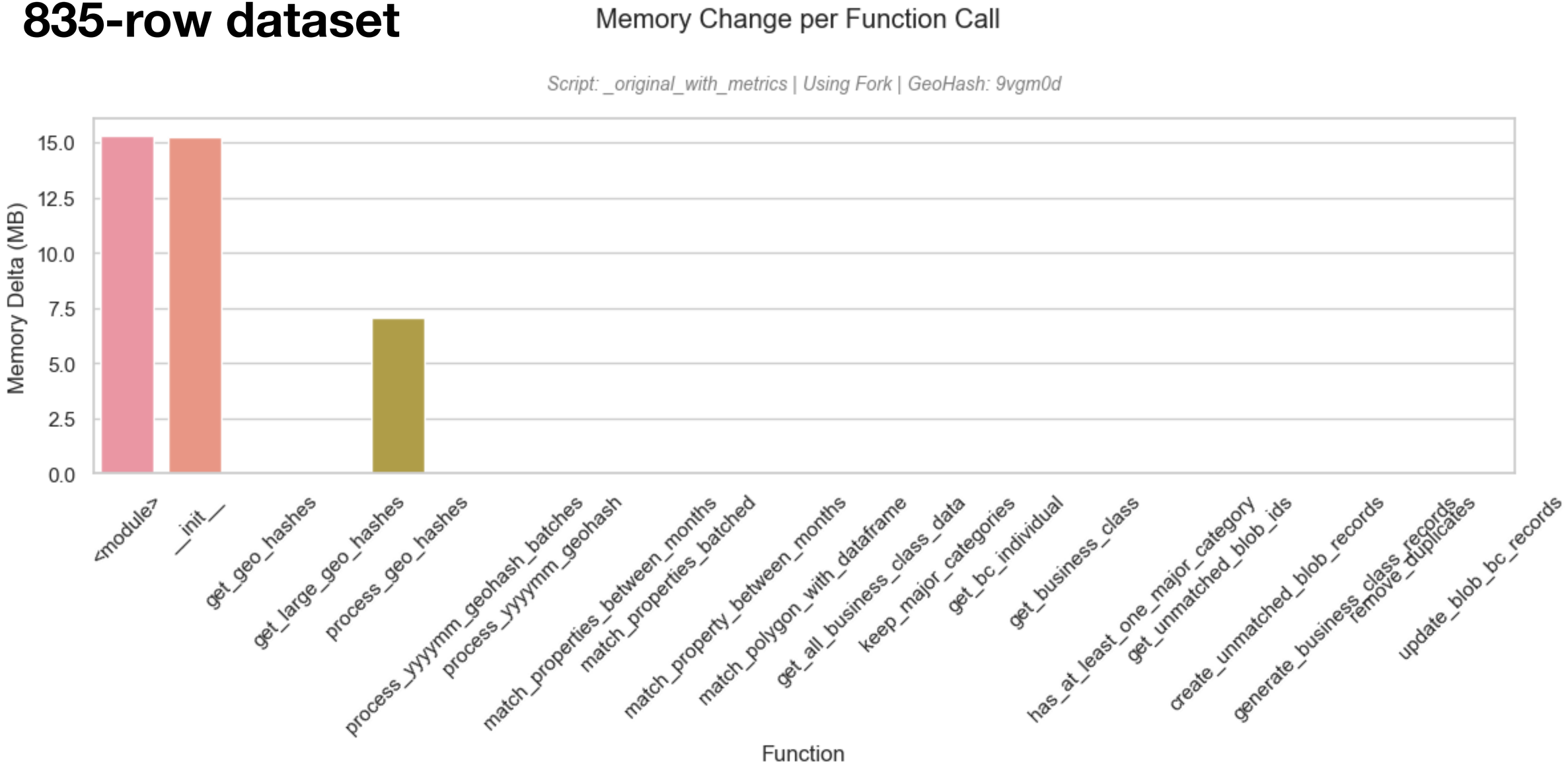
835-row dataset





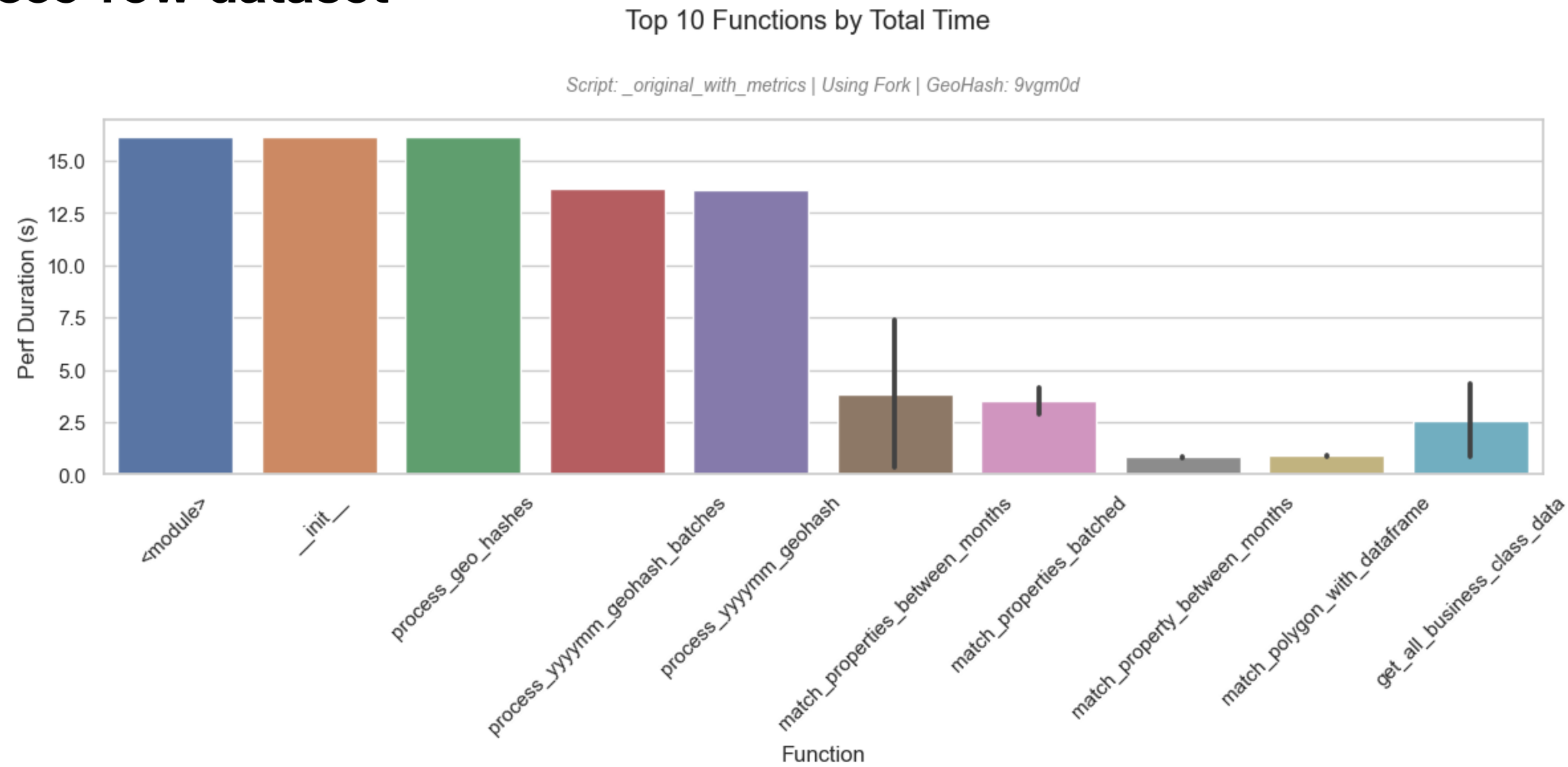
Performance Results

835-row dataset



Performance Results

835-row dataset



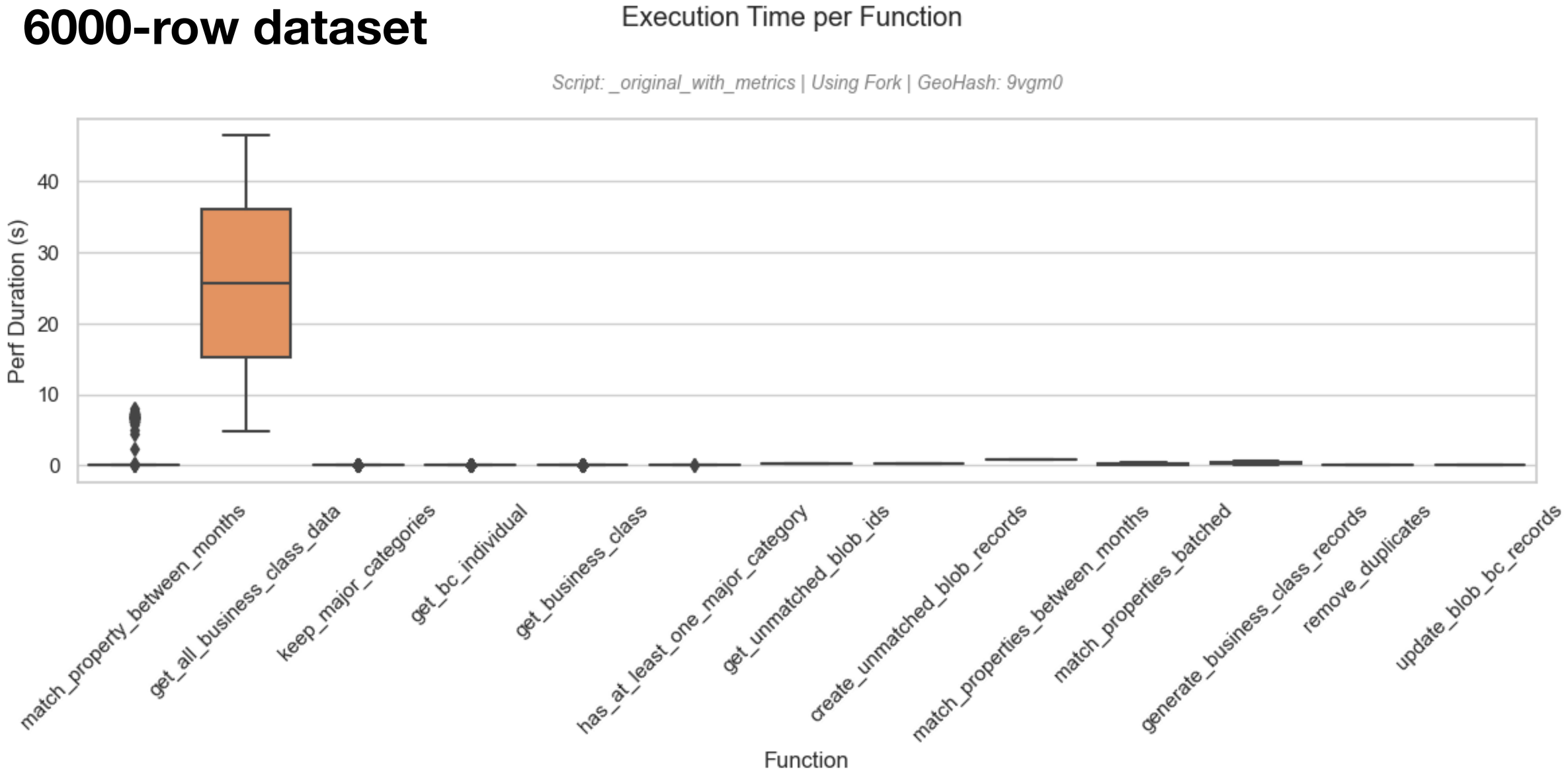
Performance Results

835-row dataset

Function	Perf Duration (s)_count	Perf Duration (s)_sum	Perf Duration (s)_mean	Perf Duration (s)_max	CPU Delta_mean	Memory Delta (MB)_mean
match_properties_batched	105	370.4236	3.5278	7.1972	0.00	0.0000
match_property_between_months	438	364.2335	0.8316	1.7845	0.00	0.0000
match_polygon_with_dataframe	389	349.6549	0.8989	1.7670	0.00	0.0000
<module>	1	16.1637	16.1637	16.1637	78.40	15.3354
__init__	1	16.1550	16.1550	16.1550	65.40	15.2699
process_geo_hashes	1	16.1157	16.1157	16.1157	-14.90	7.0451
process_yyyymm_geohash_batches	1	13.6529	13.6529	13.6529	-1.30	0.0000
process_yyyymm_geohash	1	13.5878	13.5878	13.5878	0.00	0.0000
match_properties_between_months	2	7.7221	3.8611	7.3957	0.00	0.0000
get_all_business_class_data	2	5.1677	2.5839	4.3149	0.00	0.0000
get_bc_individual	438	2.0380	0.0047	0.0831	-0.05	0.0000
keep_major_categories	438	0.6930	0.0016	0.0566	0.00	0.0000
get_business_class	403	0.4381	0.0011	0.0340	-0.05	0.0000
generate_business_class_records	2	0.1697	0.0848	0.1554	0.00	0.0000
create_unmatched_blob_records	1	0.1089	0.1089	0.1089	0.00	0.0000
get_unmatched_blob_ids	1	0.0162	0.0162	0.0162	0.00	0.0000
has_at_least_one_major_category	5	0.0045	0.0009	0.0012	0.00	0.0000
remove_duplicates	1	0.0019	0.0019	0.0019	75.10	0.0000
update_blob_bc_records	1	0.0003	0.0003	0.0003	6.80	0.0000
get_geo_hashes	1	0.0002	0.0002	0.0002	29.80	0.0000
get_large_geo_hashes	1	0.0001	0.0001	0.0001	-0.20	0.0000

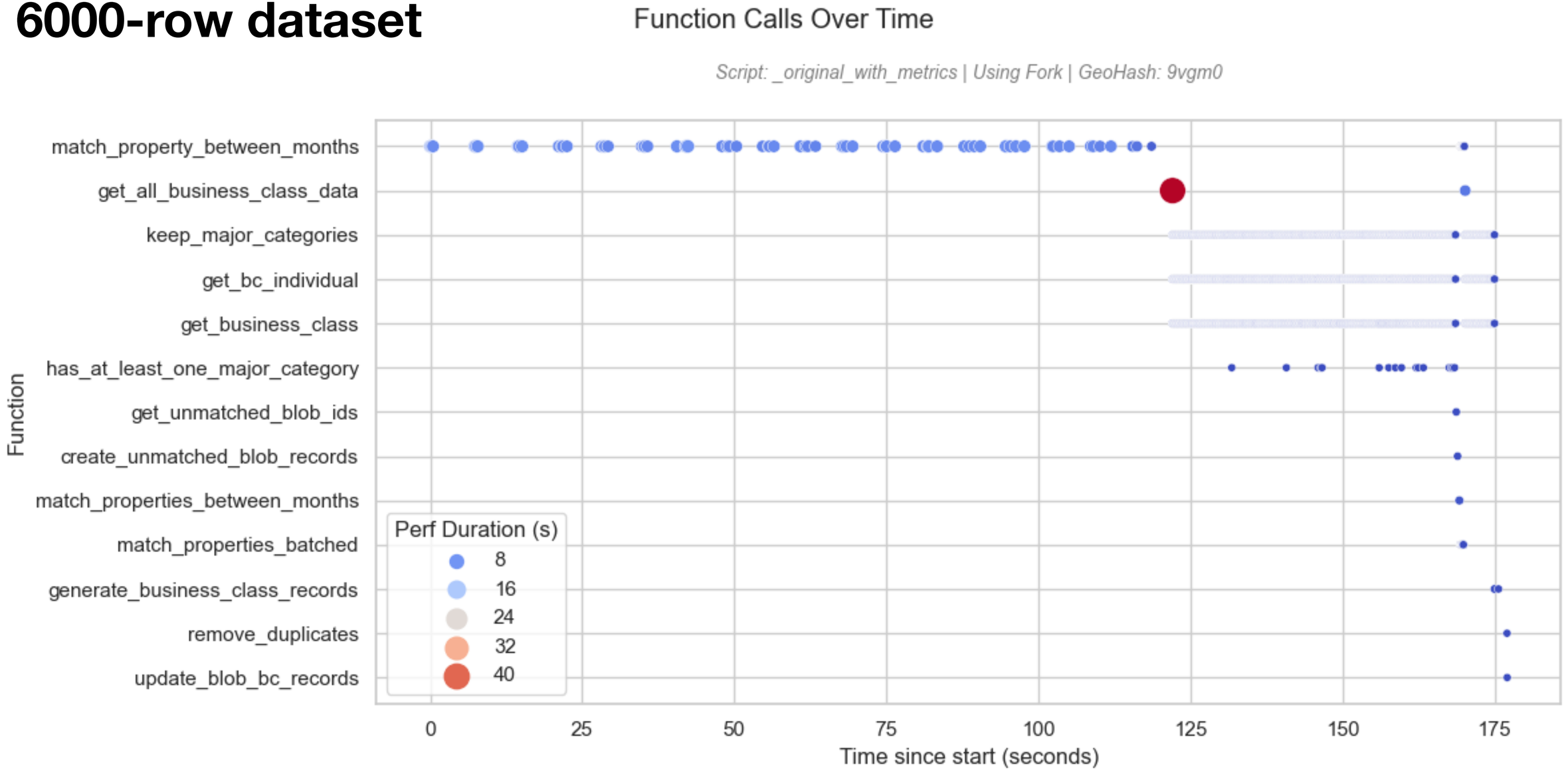
Performance Results

6000-row dataset



Performance Results

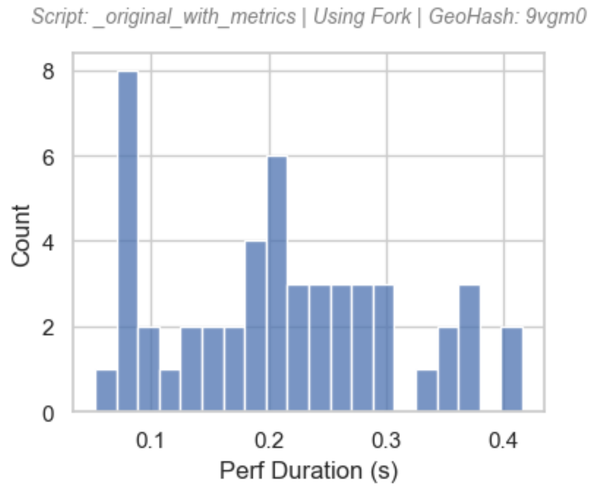
6000-row dataset



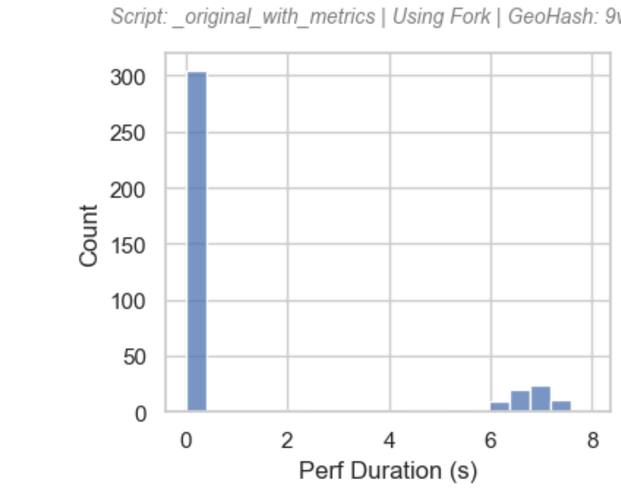
Performance Results

6000-row dataset

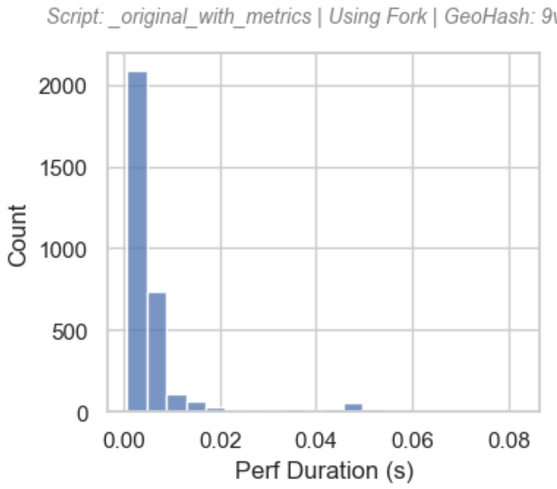
Perf Duration for 'match_properties_batched'



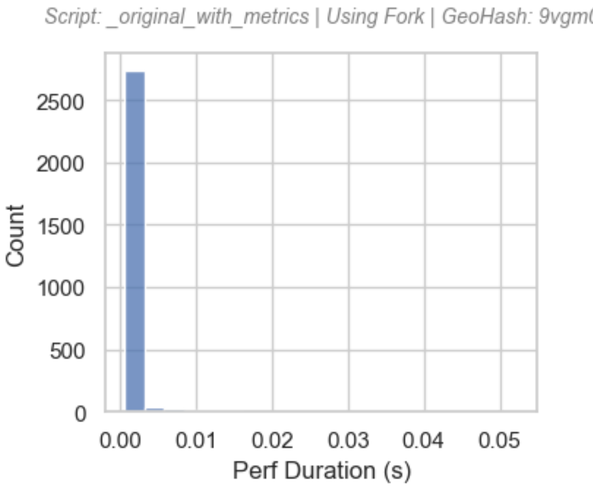
Perf Duration for 'match_property_between_months'



Perf Duration for 'get_bc_individual'

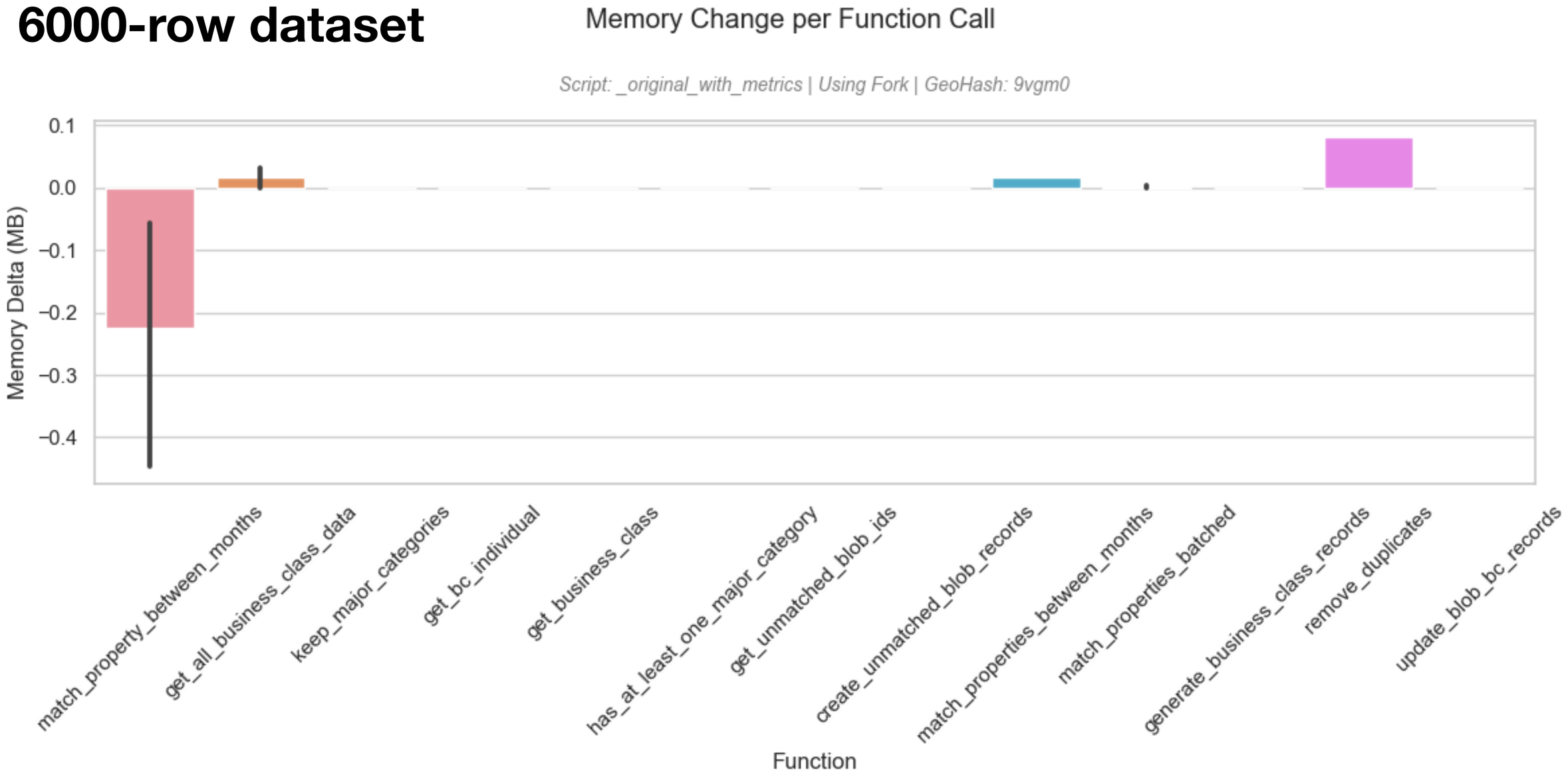


Perf Duration for 'get_business_class'



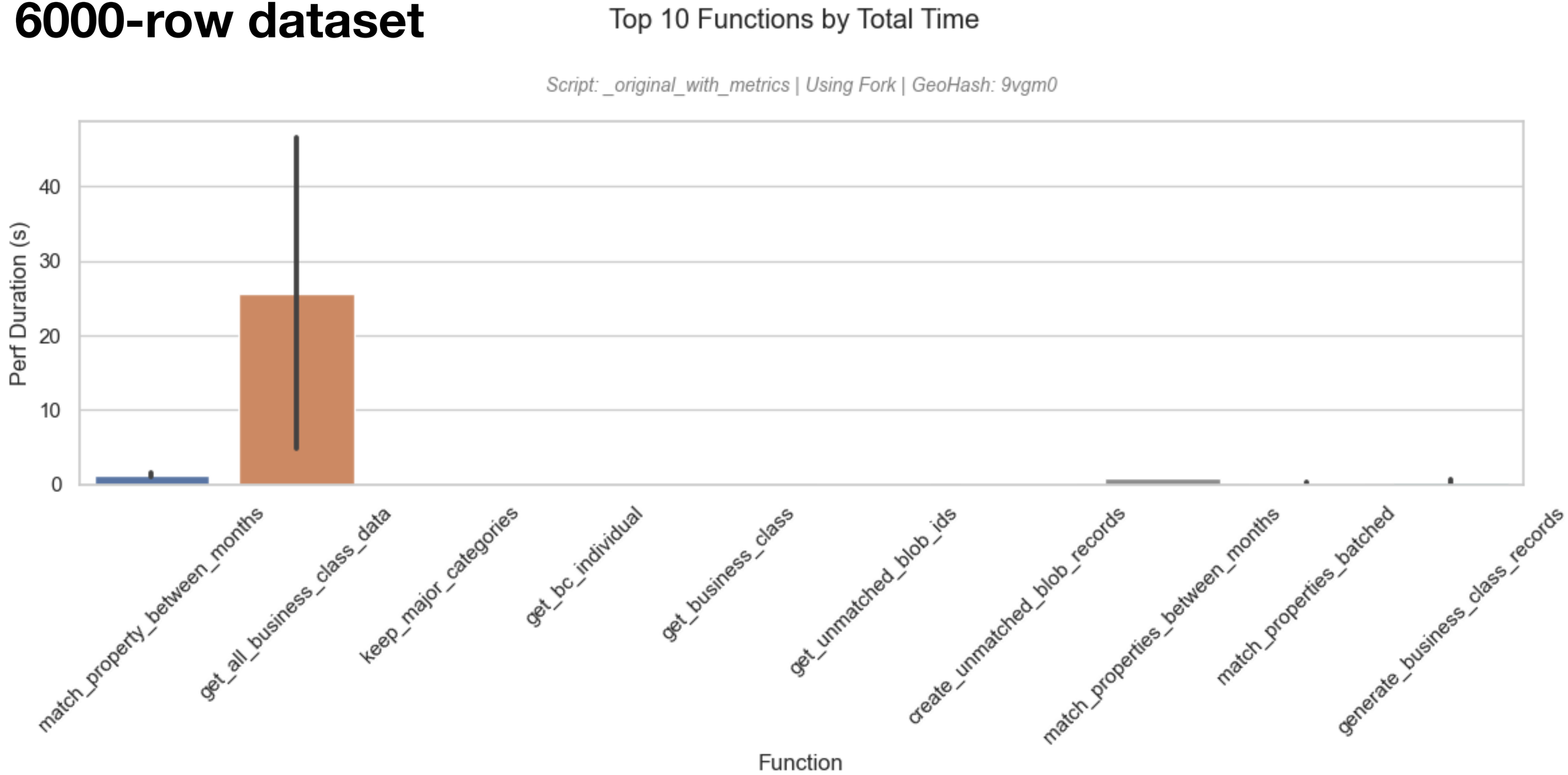
Performance Results

6000-row dataset



Performance Results

6000-row dataset



Performance Results

6000-row dataset

Function	Perf Duration (s)_count	Perf Duration (s)_sum	Perf Duration (s)_mean	Perf Duration (s)_max	CPU Delta_mean	Memory Delta (MB)_mean
match_property_between_months	376	480.7318	1.2785	7.9774	-0.0011	-0.2240
get_all_business_class_data	2	51.3464	25.6732	46.5252	0.0000	0.0164
get_bc_individual	3180	21.3697	0.0067	0.0824	0.0017	0.0000
match_properties_batched	51	10.7770	0.2113	0.4164	0.0039	0.0019
keep_major_categories	3180	6.7290	0.0021	0.0677	-0.0015	0.0000
get_business_class	2875	5.2438	0.0018	0.0524	0.0000	0.0000
match_properties_between_months	1	0.9452	0.9452	0.9452	0.0000	0.0164
generate_business_class_records	2	0.7380	0.3690	0.6396	0.0000	0.0000
create_unmatched_blob_records	1	0.2242	0.2242	0.2242	0.0000	0.0000
get_unmatched_blob_ids	1	0.2112	0.2112	0.2112	0.0000	0.0000
has_at_least_one_major_category	16	0.0232	0.0015	0.0066	0.0000	0.0000
remove_duplicates	1	0.0114	0.0114	0.0114	72.9000	0.0819
update_blob_bc_records	1	0.0006	0.0006	0.0006	67.0000	0.0000

Agenda

1. Introduction
2. Work Timeline
3. Performance Results
4. **Next Steps**

Next Steps

Key Expensive Functions and Methods

1. `match_properties_batched`
2. `match_property_between_months`
    2.1. `match_polygon_with_dataframe`
3. `process_yyyymm_geohash_batches`
4. `process_yyyymm_geohash`
5. `process_geo_hashes`
6. `__init__`

Next Steps

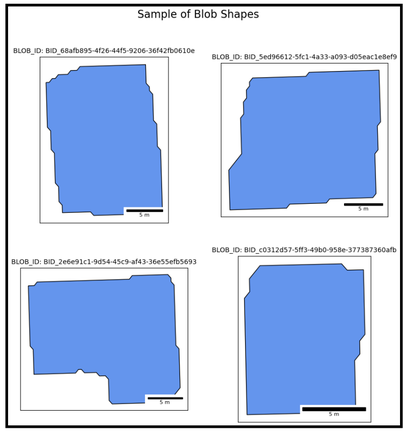
Observations and Recommendations

- Blob shapes are complicated

Next Steps

Observations and Recommendations

- Blob shapes are complicated



Next Steps

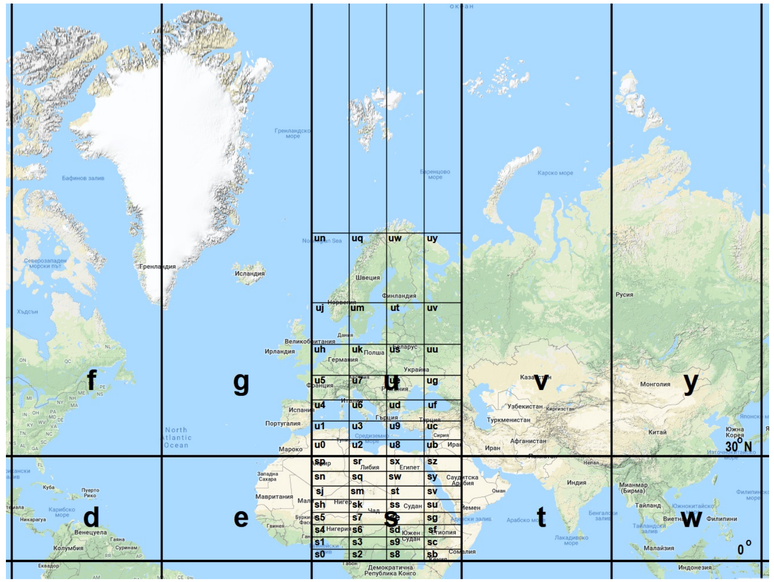
Observations and Recommendations

- Blob shapes are complicated
- Repeated communications with external database is slow

Next Steps

Observations and Recommendations

- Blob shapes are complicated
- Repeated communications with external database is slow
- Geohash is a fine geospatial index, but neighbor-finding is not as strong…

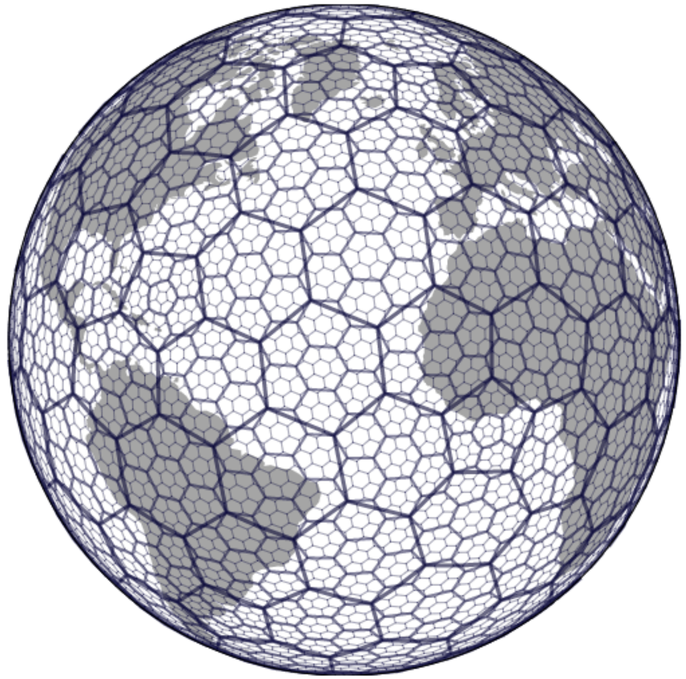


http://petrov.free.bg/academic/publication/geohash-eas-modified-geohash-geocoding-system-equal-area-spaces/fig_6.jpg

Next Steps

Observations and Recommendations

- Blob shapes are complicated
- Repeated communications with external database is slow
- Geohash is a fine geospatial index, but neighbor-finding is not as strong…
- …as other alternative options



https://viennadatasciencegroup.at/post/2019-11-21-h3spark/featured.png

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…

Next Steps

Observations and Recommendations

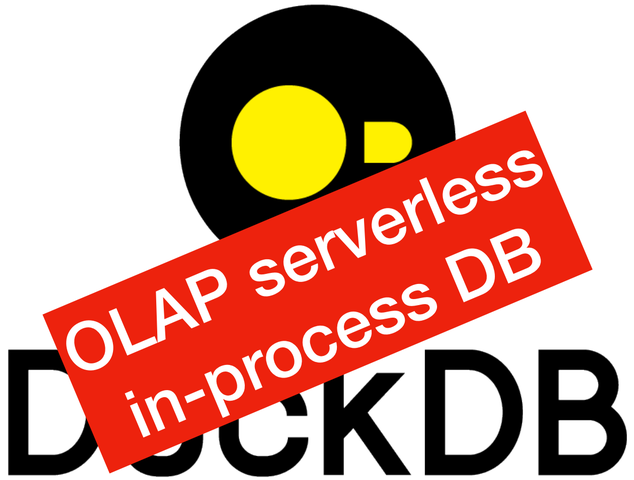
- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation

https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation

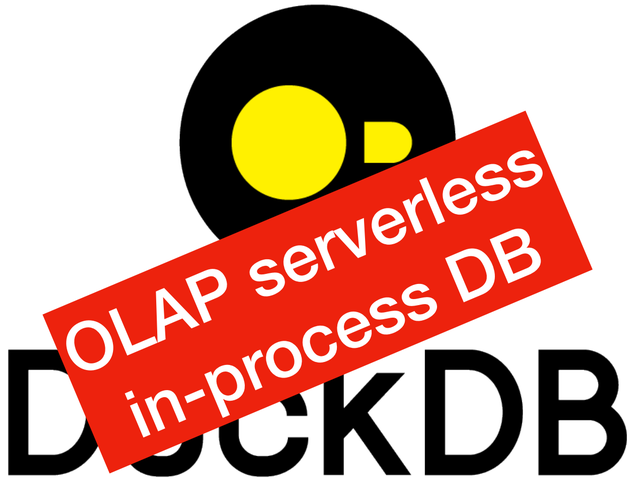


https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation



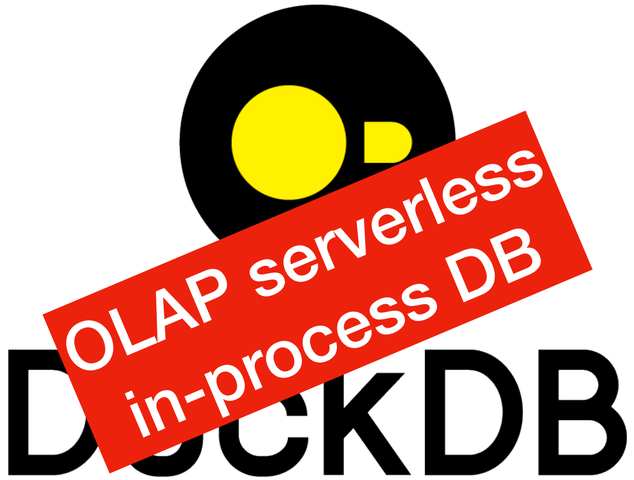
https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

https://postgis.net/

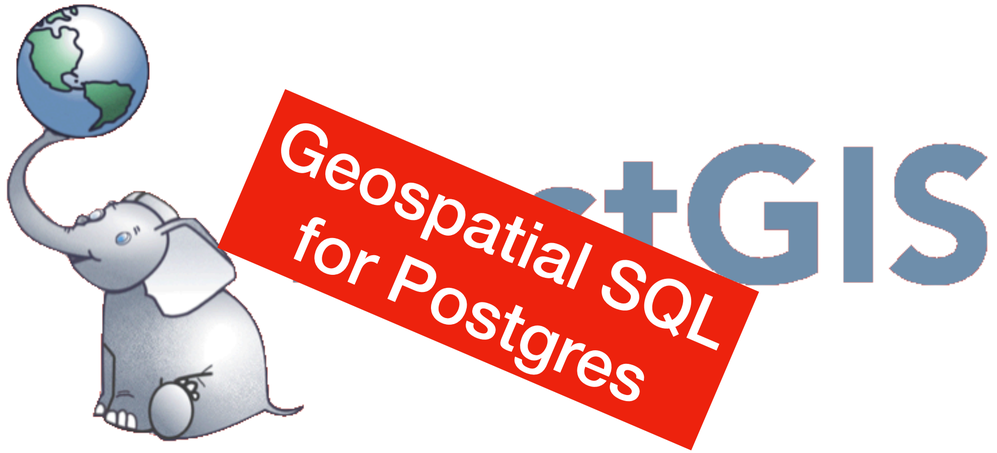
Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation



https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/



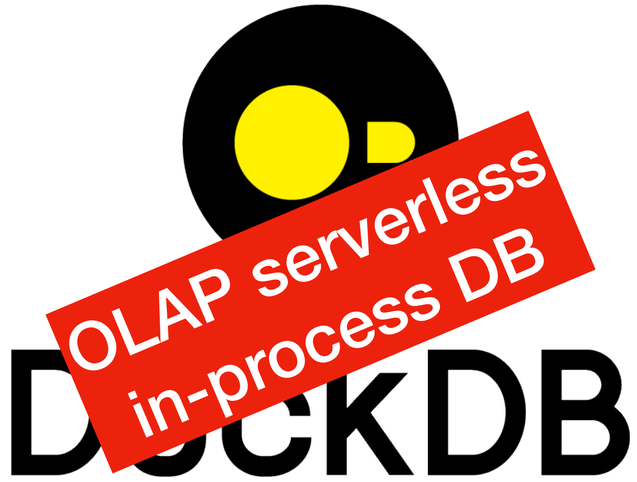
https://postgis.net/

Next Steps

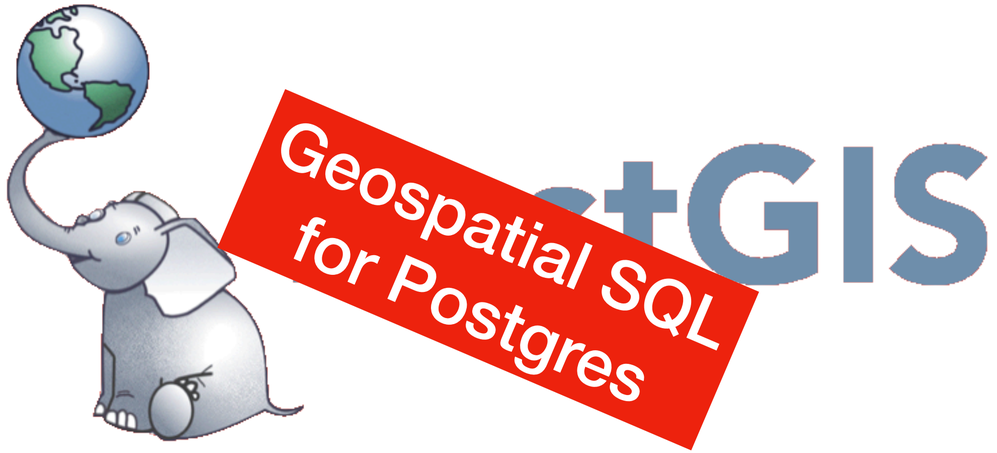
Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation

https://sedona.apache.org/latest-snapshot/api/rdocs/logo.png



https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/



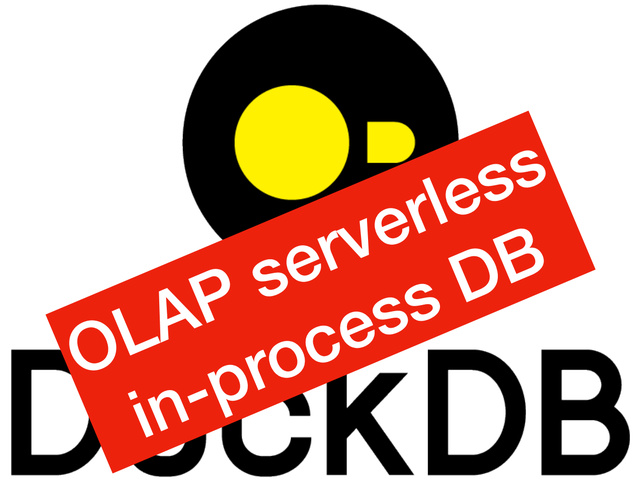
https://postgis.net/

Next Steps

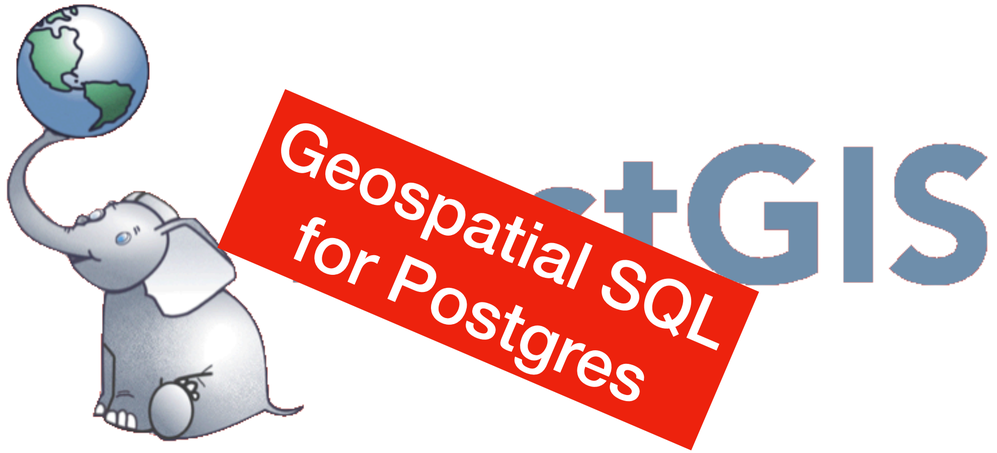
Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation

https://sedona.apache.org/latest-snapshot/api/rdocs/logo.png



https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

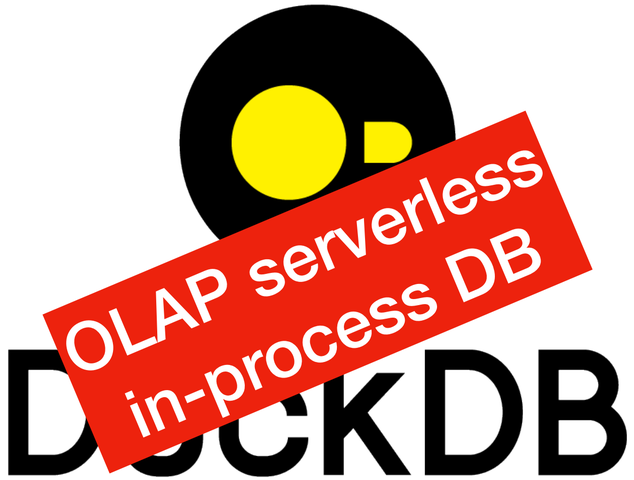


https://postgis.net/

Next Steps

Observations and Recommendations

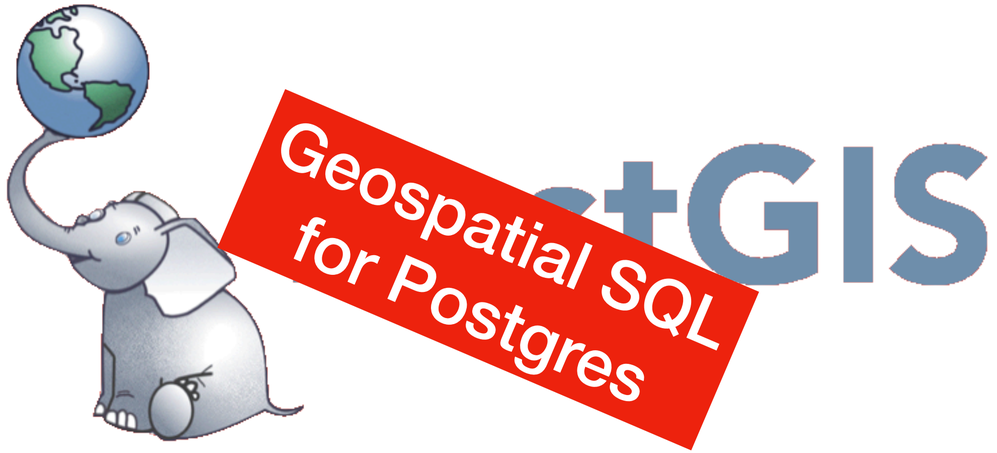
- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation



https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

https://sedona.apache.org/latest-snapshot/api/rdocs/logo.png

https://wherobots.com/

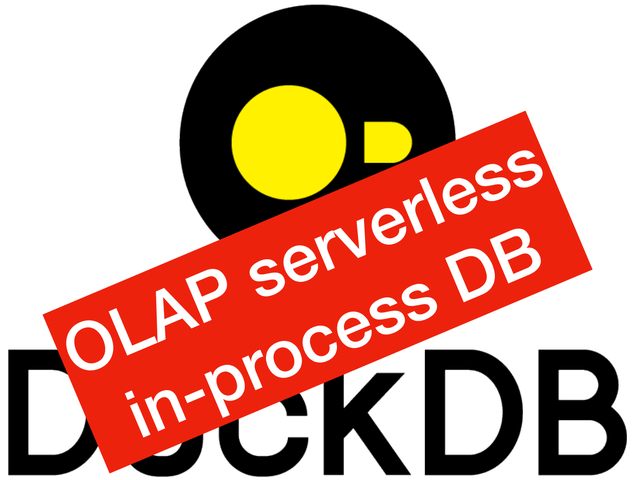


https://postgis.net/

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation



OLAP serverless in-process DB

https://www.casd.eu/en/data-tech-webinaire-parquet-duckdb/duckdb-logo-2/

Distributed geospatial processing on Spark

https://sedona.apache.org/latest-snapshot/api/rdocs/logo.png

Cloud-native geospatial data platform

Geospatial SQL for Postgres

PostGIS

https://postgis.net/

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation
- Make sure to use combination of:

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation
- Make sure to use combination of:
  - Partitioning of data

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation
- Make sure to use combination of:
  - Partitioning of data
  - Pre-filtering

Next Steps

Observations and Recommendations

- Geopandas is a fine tool for spatial join processes…
- …but at the scale we're working at, other tools merit experimentation
- Make sure to use combination of:
  - Partitioning of data
  - Pre-filtering
  - `.parquet` and other optimized data formats

Thanks!

https://github.com/sralter/pymaap