# How simple can mass-spec files get?

## Databases are a speedy, small, and simple solution for MS data storage and access

**William Kumler and Anitra E. Ingalls**
**wkumler@uw.edu or github.com/wkumler/mzsql**
**University of Washington, School of Oceanography, Seattle, USA**

### Column-based tidy format is simple and intuitive for all MS types

**Convert *m/z* and intensity tuples into database columns**

**Pair with other separation data**
- Retention time (liquid chromatography)
- Drift time (ion mobility)
- X/Y coordinate (imaging MS)

**Link with MS$^n$ data via scan number**

**Pair with filename to aggregate multifile**
- Optimized *across* files
- Metadata tables saved alongside

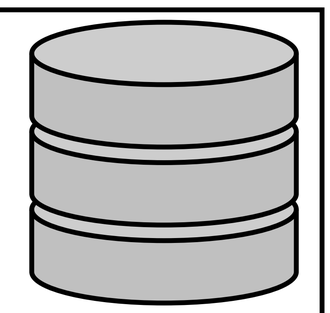
Table: MS1

| filename<br>Source file | scan_num<br>Scan number | rt<br>Retention time | mz<br>m/z ratio | int<br>Intensity |
|---|---|---|---|---|
| Smp_A | 1 | 0.10 | 60.0452 | 6618 |
| Smp_A | 1 | 0.10 | 60.0532 | 2657 |
| ...millions of additional entries... | | | | |
| Smp_Z | 1385 | 22.35 | 60.0456 | 158084 |
| Smp_Z | 1385 | 22.35 | 60.0531 | 4673 |

Table: MS2

| filename<br>Source file | scan_num<br>Scan number | prescan<br>Precursor scan | rt<br>Retention time | fragmz<br>Fragment m/z | premz<br>Precursor m/z | int<br>Intensity |
|---|---|---|---|---|---|---|
| Smp_A | 2 | 1 | 0.12 | 51.0238 | 241.0894 | 36104 |
| Smp_A | 2 | 1 | 0.12 | 53.0394 | 241.0894 | 243165 |
| ...millions of additional entries... | | | | | | |
| Smp_Z | 1390 | 1385 | 22.45 | 52.0186 | 185.1932 | 28371 |
| Smp_Z | 1390 | 1385 | 22.45 | 57.0923 | 185.1932 | 129604 |

Ion chromatogram extraction:
```
SELECT * FROM MS1 WHERE mz BETWEEN min AND max
```
Retention time range subset:
```
SELECT * FROM MS1 WHERE rt BETWEEN min AND max
```
Fragment search:
```
SELECT * FROM MS2 WHERE fragmz BETWEEN min AND max
```
Precursor search:
```
SELECT * FROM MS2 WHERE premz BETWEEN min AND max
```



**Database**
**.sqlite**
**.duckdb**
**.parquet**

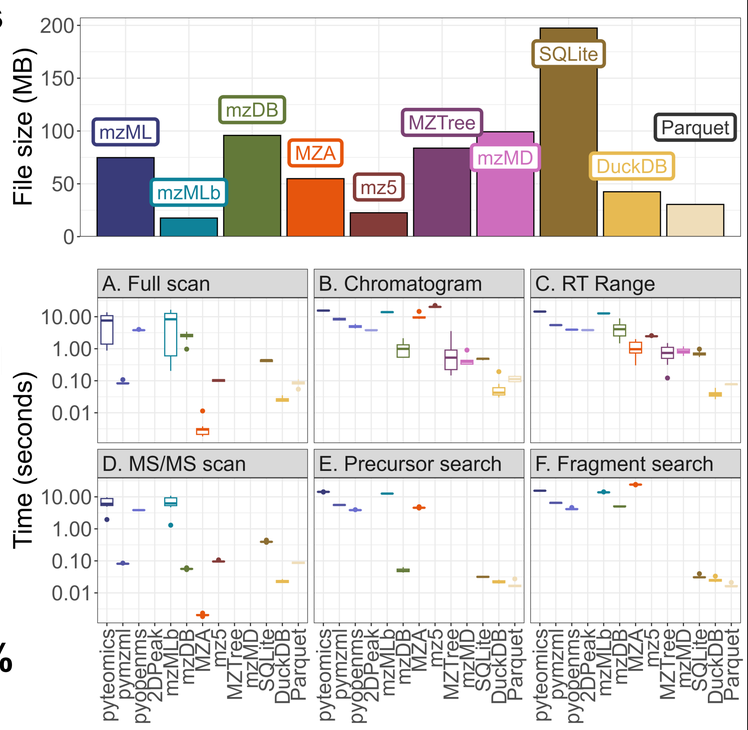
# Problem: very large in memory! Solution: simple databases

### Databases vs existing MS file options

**We compared 13 different mass-spectrometry file types in Python using six metrics**
- Single MS$^1$ scan
- Ion chromatogram
- Consecutive MS$^1$ scans
- Single MS$^2$ scan
- Precursor *m/z* search
- Fragment *m/z* search

**Databases outperformed on all metrics other than known scan extraction**
- Often by 10-100x

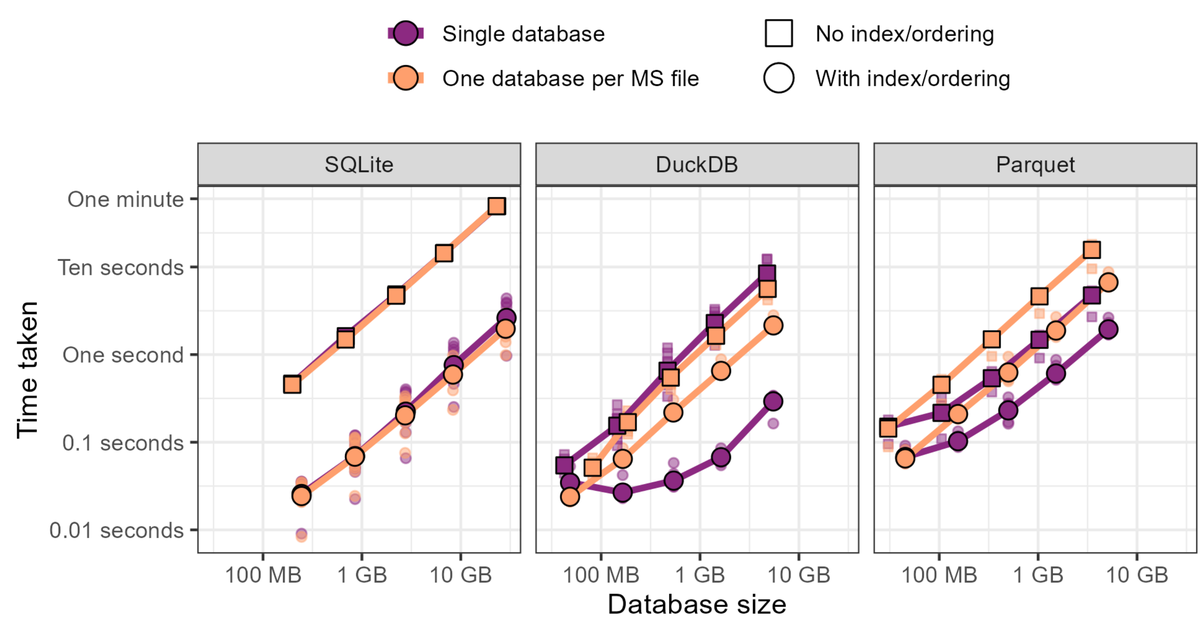
**Parquet and DuckDB methods also reduced the on-disk file size ~50%**




### Optimization: index / order / aggregate

**Indexing/ordering databases improved again by 10-100x**

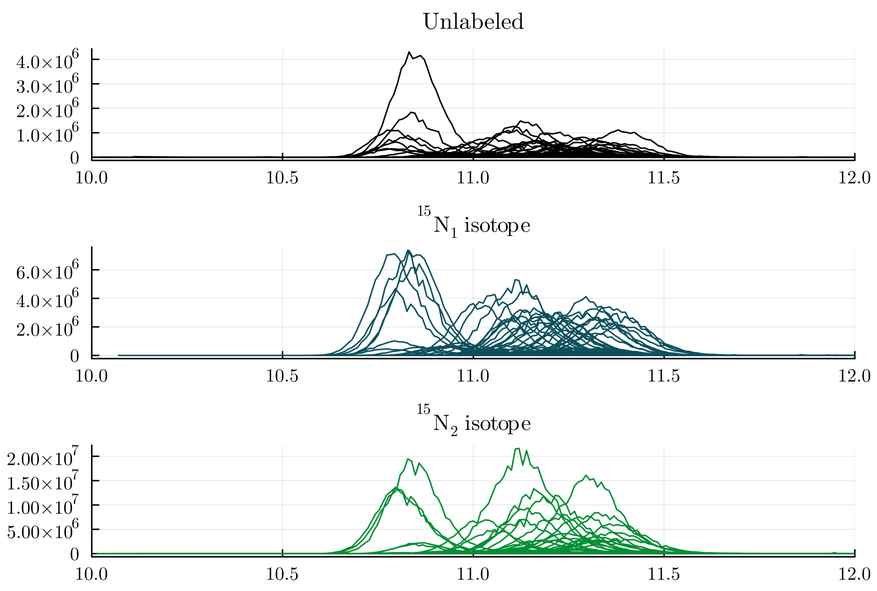
**Improvements magnified with multiple files in one database**




# Databases offer language agnostic data access and visualization with essentially zero memory overhead

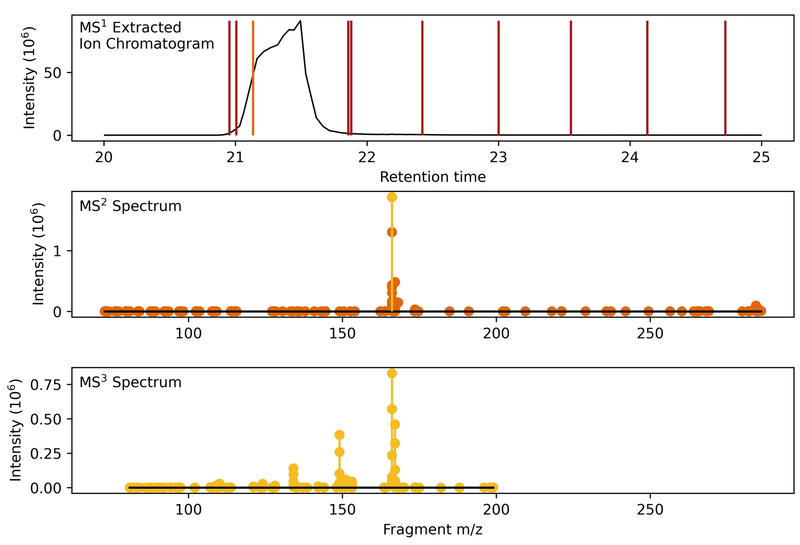
### EICs in Julia via SQLite

```julia
using SQLite, DataFrames, StatsPlots

db = SQLite.DB("chrom_data/chrom_data.sqlite")
targets = [147.0764, 148.0734, 149.0705]
window(m) = (m*(1 - 2/1e6), m*(1 + 2/1e6))
clause = join(["(mz BETWEEN $(mzmin) AND $(mzmax))" for (mzmin, mzmax) in window.(targets)], " OR ")
query = "SELECT * FROM MS1 WHERE ($clause) AND (rt BETWEEN 10 AND 12)"
df = DBInterface.execute(db, query) |> DataFrame
SQLite.close(db)

p1 = @df df[round.(df.mz) .== 147, :] plot(:rt, :int, group=:filename, color="black", title="Unlabeled")
p2 = @df df[round.(df.mz) .== 148, :] plot(:rt, :int, group=:filename, color="#0b505c", title="15Nx1")
p3 = @df df[round.(df.mz) .== 149, :] plot(:rt, :int, group=:filename, color="#028e34", title="15Nx2")
plot(p1, p2, p3, layout = (3,1), dpi=600)
```



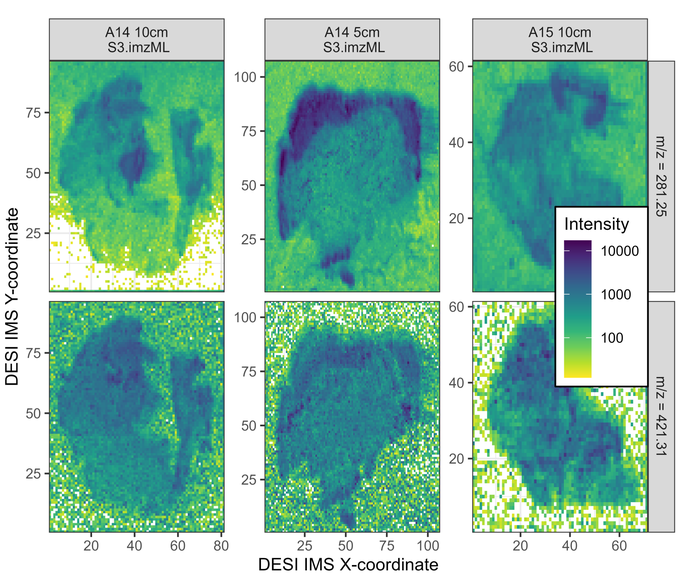

### MS$^3$ in Python via DuckDB

```python
import duckdb, matplotlib.pyplot as plt, pandas as pd

con = duckdb.connect("ms3_data/ms3_data.duckdb")
ms1 = con.execute("SELECT * FROM MS1 WHERE mz BETWEEN 282.1186 AND 282.1208").fetchdf()
ms2 = con.execute("SELECT * FROM MS2 WHERE premz BETWEEN 282.1186 AND 282.1208").fetchdf()
ms3 = con.execute("""SELECT * FROM MS3 WHERE prepremz BETWEEN 282.1186 AND 282.1208 AND
                     premz BETWEEN 166.0718 AND 166.0731""").fetchdf()
con.close()

fig, axs = plt.subplots(3, 1, figsize=(6, 4))
axs[0].plot(ms1['rt'], ms1['int'])
axs[0].vlines(ms2['rt'], ymin=0, ymax=ms1['int'].max())
axs[1].stem(ms2['fragmz'], ms2['int'])
axs[2].stem(ms3['fragmz'], ms3['int'])
plt.savefig("ms3_duckdb_python_fig.png", dpi=600, bbox_inches="tight")
```




### Imaging in R via Parquet

```r
library(tidyverse)
library(RaMS)
arrow::open_dataset("imzml_data/pqds") %>%
  filter(mz%between%pmppm(421.3111, 10) | mz%between%pmppm(281.2484, 10)) %>%
  mutate(mz=paste("m/z =", round(mz, 2))) %>%
  collect() %>%
  ggplot(aes(x, y, fill=int)) + geom_raster() + facet_grid(mz~filename)
```




# Simpler is often smaller *and* speedier