# Moving Weather Model Ensembles To A Geospatial Database For The Aviation Weather Center

Presented by Jeff Smith

NOAA's Earth System Research Lab in Boulder, CO

April 2, 2013

## Background -1

- NCEP – is NOAA / NWS / National Centers for Environmental Prediction
- AWC – is NOAA / NWS / Aviation Weather
- An ensemble is a collection of individual forecasts that can be averaged in some way—the idea being that the collection tends to predict the weather better than one individual forecast.
- Ensembles naturally lend themselves to probabilistic forecasting
  - For example, if 5 out of 20 forecasts predict rain at a particular location, you could say that the chance of rain is 25%

## Background -2

- The NCEP 40 km SREF (short range ensemble forecast) is run/updated every 6 hours
- 21 ensemble members
  - 1176 grib2 files (5.5 GB)
  - Forecast hours from 0 to 87
  - 7 WRF NMM
  - 7 WRF ARW
  - 7 NMB members
- The SREF grib2 files are FTP’d to the AWC over a period of 60 to 90 minutes as the files become available

## Background -3

- NCEP operational 40km SREF (short range ensemble forecast) covers grid 212

185 x 129 grid

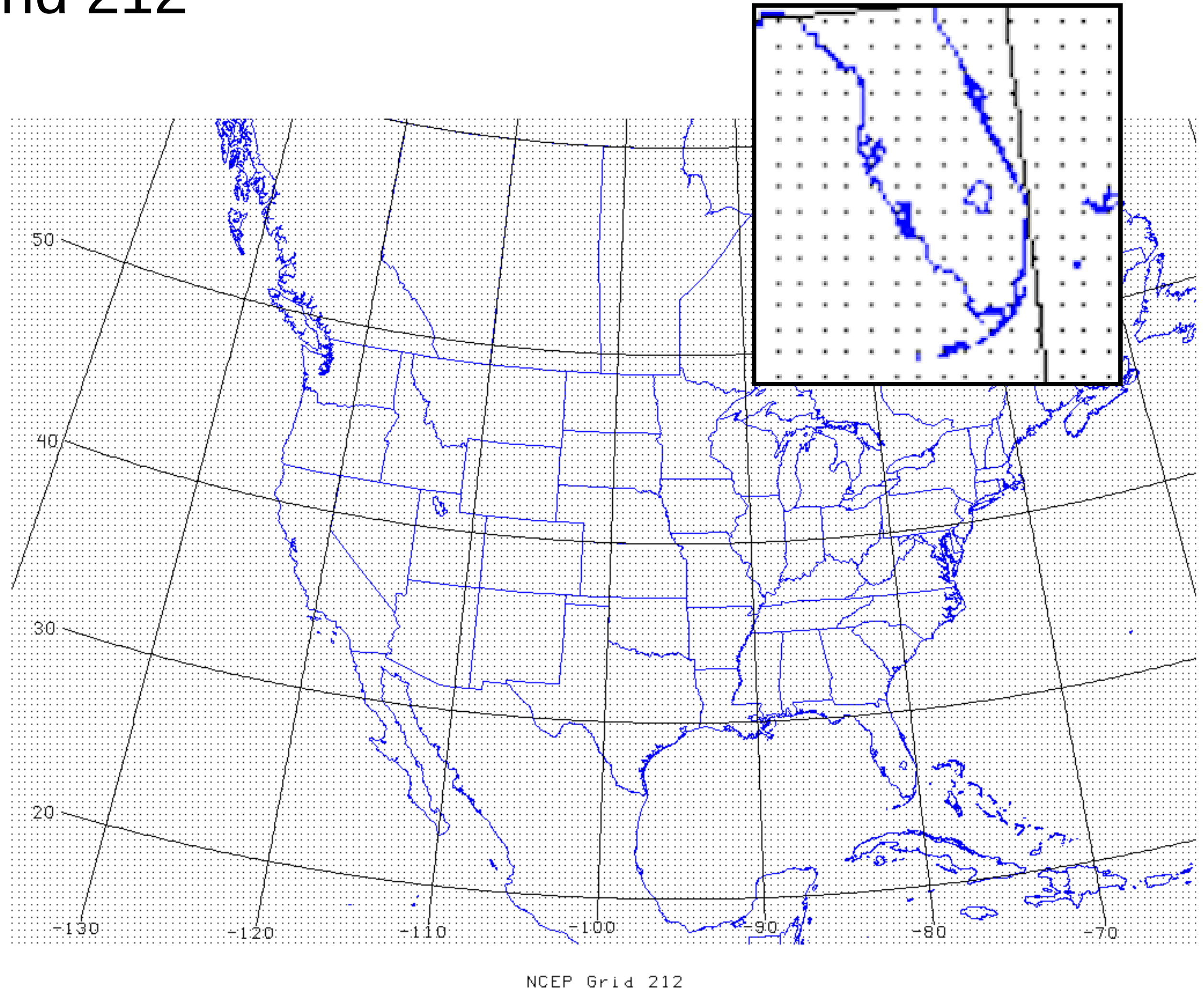
39 vertical levels

90 variables

56 output hours

21 ensemble members

1176 grib2 files



## Background -4

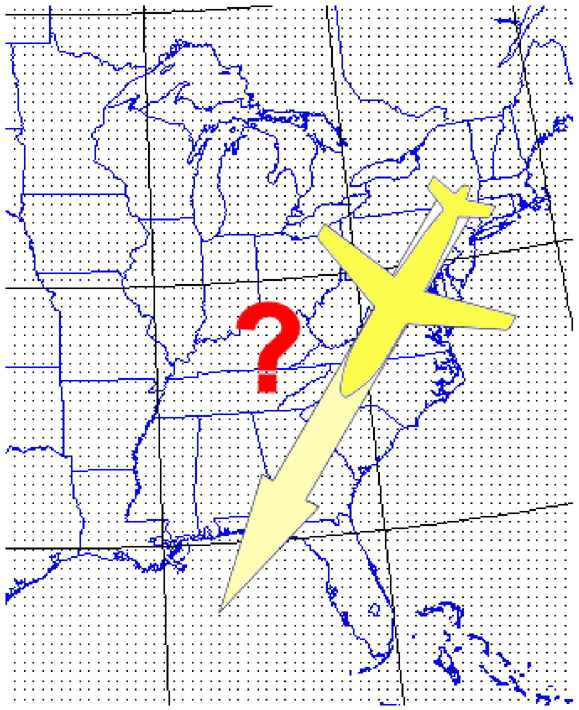
- Analyzing, aggregating, and visualizing data in multiple data files is difficult
- For example, what if you want to answer this question:

  "What is the maximum wind between 250-500 mb over the western half of United States from forecast hour 30-35, and which ensemble member predicted the highest value?"

## Background -5

- Another question might be

  “What is the ensemble mean temperature at every grid point at 950mb within an air corridor along the East Coast at forecast hour 10?”



## Background -6

- To answer questions such as these, a forecaster might write a computer program that
  - opens and reads each of these 1176 files
  - potentially writes the intermediate results to some large intermediate file (if it won’t fit into memory)
  - scans this intermediate file to build an array of data values, and then writes these data values back to disk in the form of a grib2 file
- This is a time-consuming process involving developing and testing complicated program code, and the run time performance can be poor.

## Geospatial Databases

- build on standard relational databases (same tables, columns, keys, etc.)
- efficiently store data that is in geospatial (e.g. latitude and longitude) coordinates
- understand coordinate systems (and transformations)
- understand points, lines, and polygons
  - For example, instead of subsetting data within a simple bounding box (4 coordinates), you could subset your data within a complex polygon (for example, an air corridor).
- Oracle Geospatial is very good (and also expensive)
- PostGIS is built on top of Postgres and is free
- MySQL has geospatial extensions and is also free

## PostGIS versus MySQL -1

- I experimented with both PostGIS and MySQL
- PostGIS
  - More complete geospatial support than MySQL (close to Oracle)
  - Big open source community
  - Faster for some geospatial queries, better at leveraging secondary indexes
  - Creating indexes is slower than MySQL/MyISAM
  - Used psql COPY command to bulk load tens of thousands of records per second into the database. Still takes about one hour and a half to load the 1176 files (billions of records) into the database
  - Tweaked the postgresql.conf file for maximum performance with this mostly read-only database
  - Database is very large: 192 GB when 1176 files have been imported into it. This size is problematic, particularly when we move to smaller (but faster) solid state drives.

## PostGIS versus MySQL -2

- MySQL (MyISAM engine)
  - Now owned by Oracle so some fear that Oracle may not invest much into a "free" database
  - Faster for full table scans (especially with fixed-row tables)
  - Used LOAD DATA sql command for bulk loading data. LOAD DATA was much faster than PSQL COPY at loading grib2 files into database (almost 300% faster). Takes about 30 minutes to load 1176 grib2 files.
  - MyISAM database engine doesn't handle a large number of concurrent transactions well since it implements table level locking (as opposed to row level locking), but this isn't an issue since there are no inserts, updates, or deletes from this database.
  - Database is "only" 100 GB (significantly smaller than PostGIS)

- The desire for a quicker load time and a smaller database trumped the PostGIS advantages, so I went with MySQL (MyISAM engine)

## Bulk Import Process

- I’m using the NetCDF Java library (thank you, Unidata) to read the grib2 files
- My Java program spawns multiple threads, each of which reads its group of grib2 files, writes intermediate CSV files (comma separated value), then does the bulk load via the LOAD DATA MySQL statement.
- To determine which variables to read and for which forecast hours, it reads an xml configuration file on startup
- It runs in real-time. Once it is done importing a SREF date/time, it sleeps for a while then checks to see if new files have become available in the FTP directory

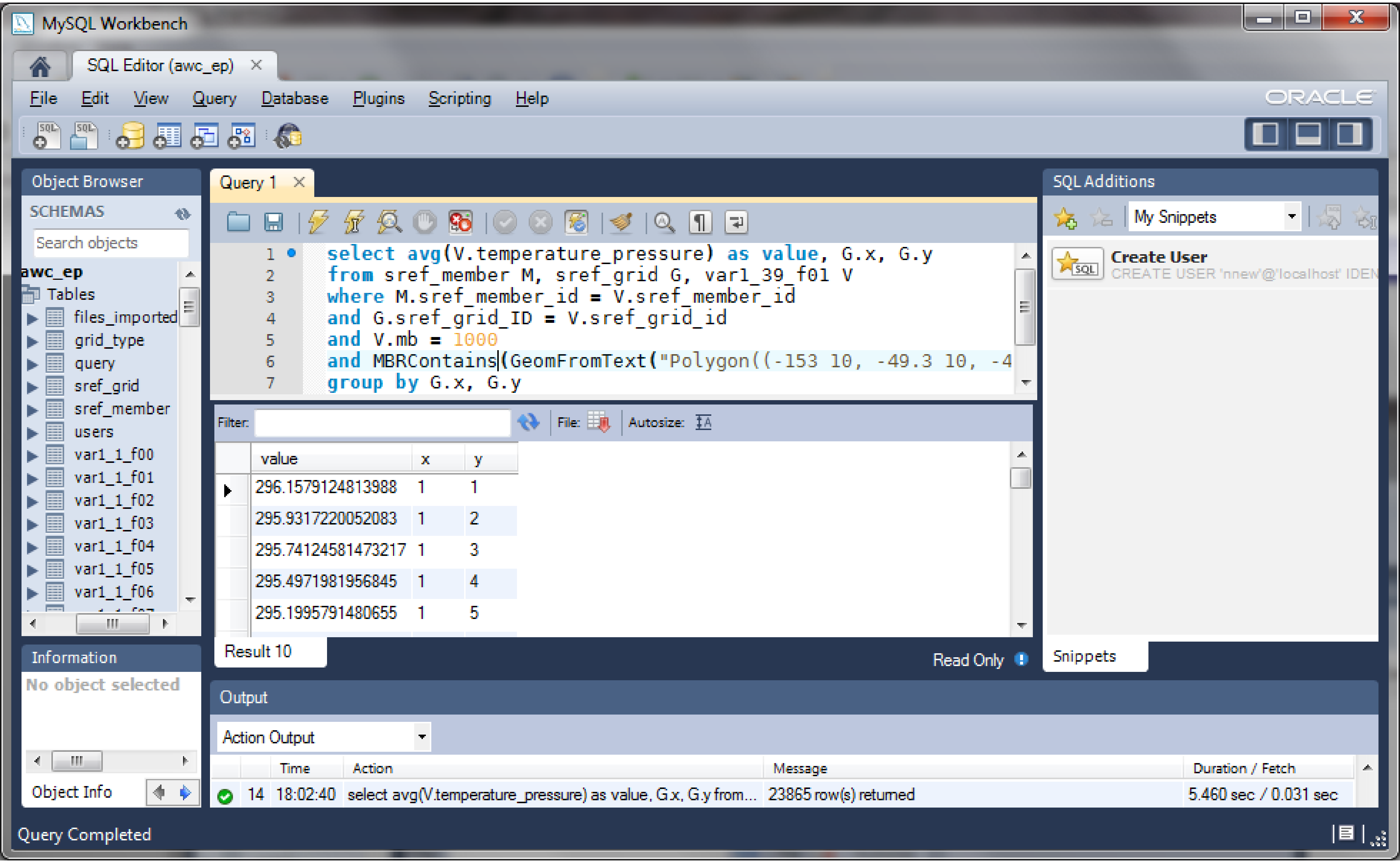
## First Sample Geospatial Query -1

This query finds the ensemble mean temperature for 21 members at 1000mb, forecast hour 01 at every grid point (23865 points) within the given bounding polygon. It runs in about 5 seconds on my laptop.

```
select avg(V.temperature_pressure) as value, G.x, G.y
from sref_member M, sref_grid G, var1_39_f01 V
where M.sref_member_id = V.sref_member_id
and G.sref_grid_ID = V.sref_grid_id
and V.mb = 1000
and MBRContains(GeomFromText("Polygon((-153 10, -49.3 10, -49.3 61.5, -153 61.5, -153 10))"), G.geom)
group by G.x, G.y
```

The MBRContains part finds points within the given geometry (which is indexed for faster access).

# MySQL Workbench -1



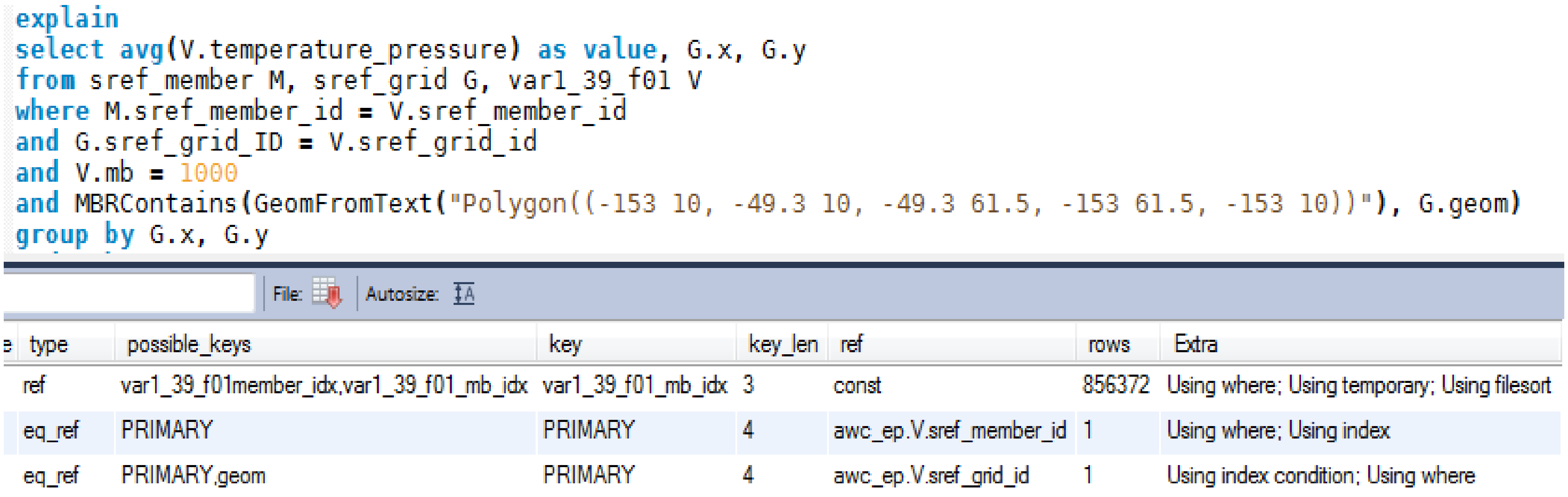
```
select avg(V.temperature_pressure) as value, G.x, G.y
from sref_member M, sref_grid G, var1_39_f01 V
where M.sref_member_id = V.sref_member_id
and G.sref_grid_ID = V.sref_grid_id
and V.mb = 1000
and MBRContains(GeomFromText("Polygon((-153 10, -49.3 10, -4
group by G.x, G.y
```

| value | x | y |
|---|---|---|
| 296.1579124813988 | 1 | 1 |
| 295.9317220052083 | 1 | 2 |
| 295.74124581473217 | 1 | 3 |
| 295.4971981956845 | 1 | 4 |
| 295.1995791480655 | 1 | 5 |

| | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|
| 14 | 18:02:40 | select avg(V.temperature_pressure) as value, G.x, G.y from... | 23865 row(s) returned | 5.460 sec / 0.031 sec |

# MySQL Workbench -2

MySQL Workbench (like pgadmin3 for Postgres/PostGIS) is a useful tool for analyzing queries and figuring out how to make them faster. Here I have done an “explain” on the query.



```
explain
select avg(V.temperature_pressure) as value, G.x, G.y
from sref_member M, sref_grid G, var1_39_f01 V
where M.sref_member_id = V.sref_member_id
and G.sref_grid_ID = V.sref_grid_id
and V.mb = 1000
and MBRContains(GeomFromText("Polygon((-153 10, -49.3 10, -49.3 61.5, -153 61.5, -153 10))"), G.geom)
group by G.x, G.y
```

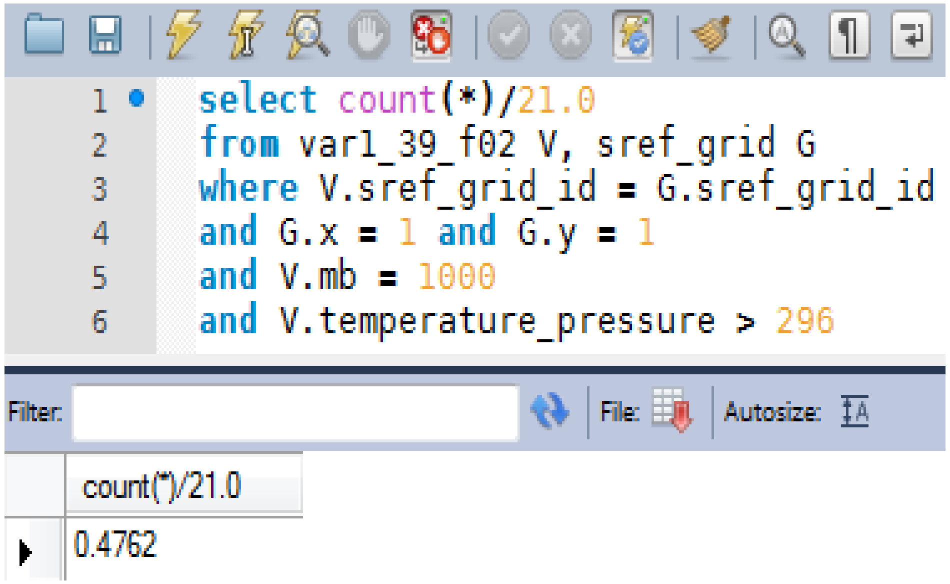
| type | possible_keys | key | key_len | ref | rows | Extra |
|---|---|---|---|---|---|---|
| ref | var1_39_f01member_idx,var1_39_f01_mb_idx | var1_39_f01_mb_idx | 3 | const | 856372 | Using where; Using temporary; Using filesort |
| eq_ref | PRIMARY | PRIMARY | 4 | awc_ep.V.sref_member_id | 1 | Using where; Using index |
| eq_ref | PRIMARY,geom | PRIMARY | 4 | awc_ep.V.sref_grid_id | 1 | Using index condition; Using where |

## Second Sample Geospatial Query

This query computes the probability that the temperature will be greater than 296K at 1000mb at one grid point.

```
select count(*)/21.0
from var1_39_f02 V, sref_grid G
where V.sref_grid_id = G.sref_grid_id
and G.x = 1 and G.y = 1
and V.mb = 1000
and V.temperature_pressure > 296
```

It returns the probability 0.4762 (47.62%) in 1.2 seconds

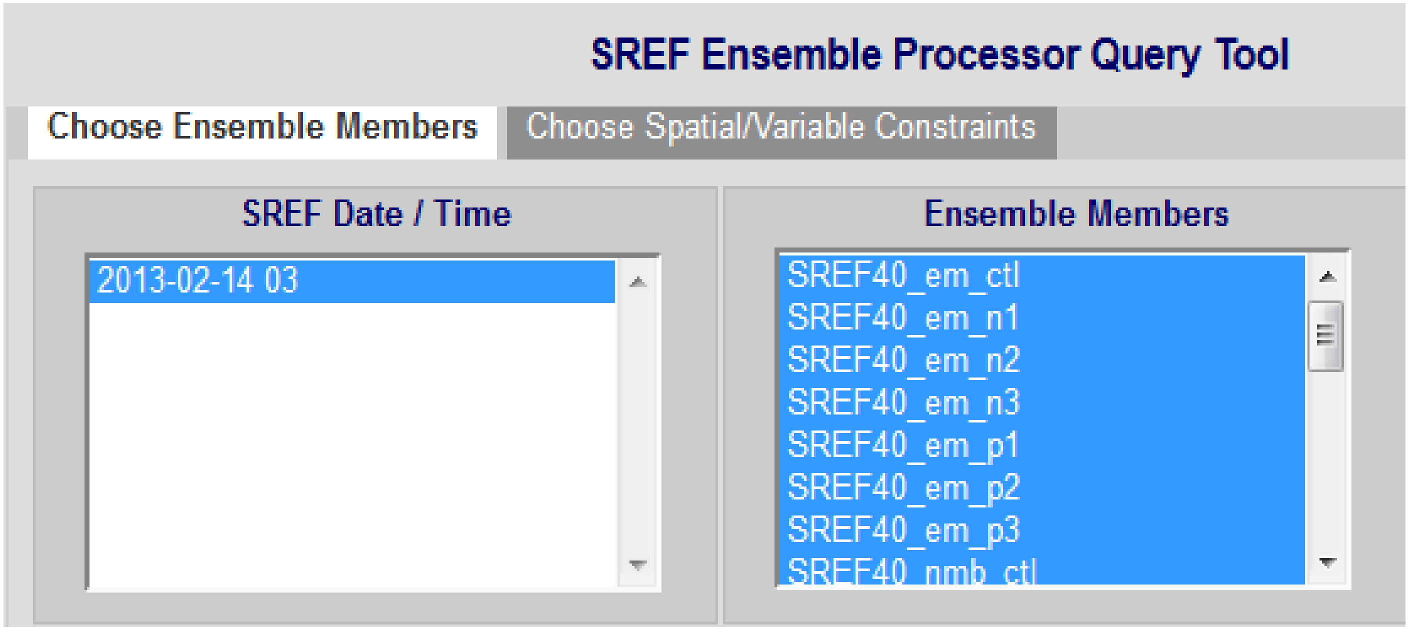


```
select count(*)/21.0
from var1_39_f02 V, sref_grid G
where V.sref_grid_id = G.sref_grid_id
and G.x = 1 and G.y = 1
and V.mb = 1000
and V.temperature_pressure > 296
```

| count(*)/21.0 |
|---|
| 0.4762 |

## SREF Query Tool -1

- To make it easier for forecasters to formulate their own queries, I created a web application in Java and Google Web Toolkit (GWT) that communicates with the MySQL database.
- The construction of queries is guided by various lists and drop down lists. The user can select the SREF date and ensemble members he is interested in.

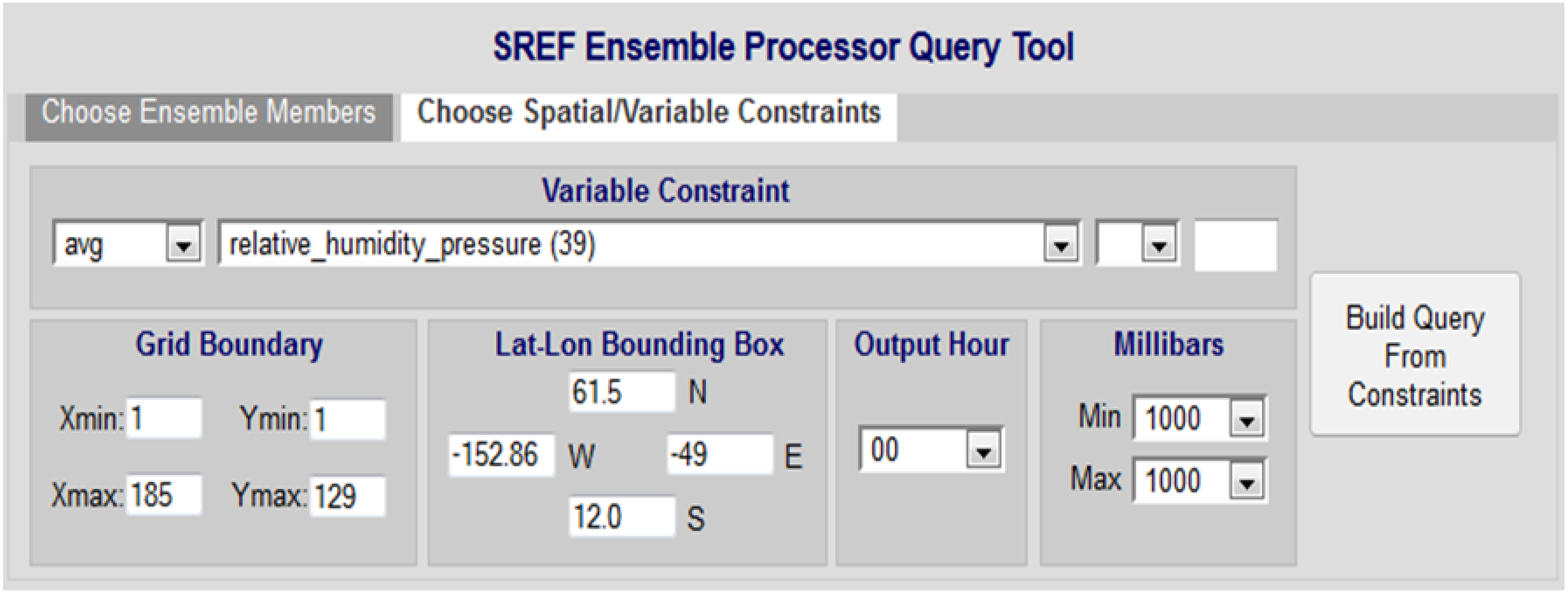

SREF Ensemble Processor Query Tool
Choose Ensemble Members
Choose Spatial/Variable Constraints
SREF Date / Time
Ensemble Members

2013-02-14 03

SREF40_em_ctl
SREF40_em_n1
SREF40_em_n2
SREF40_em_n3
SREF40_em_p1
SREF40_em_p2
SREF40_em_p3
SREF40_nmb_ctl

# SREF Query Tool -2

- Next the user can select an optional aggregate function (like a mean or standard deviation)
- The user selects a variable and an optional comparison clause (e.g. variable > 296K)
- The user can further constrain the query by geospatial region and vertical level (in millibars)



SREF Ensemble Processor Query Tool

Choose Ensemble Members | Choose Spatial/Variable Constraints

Variable Constraint

avg | relative_humidity_pressure (39)

Grid Boundary

Xmin: 1 Ymin: 1

Xmax: 185 Ymax: 129

Lat-Lon Bounding Box

61.5 N

-152.86 W -49 E

12.0 S

Output Hour

00

Millibars

Min 1000

Max 1000

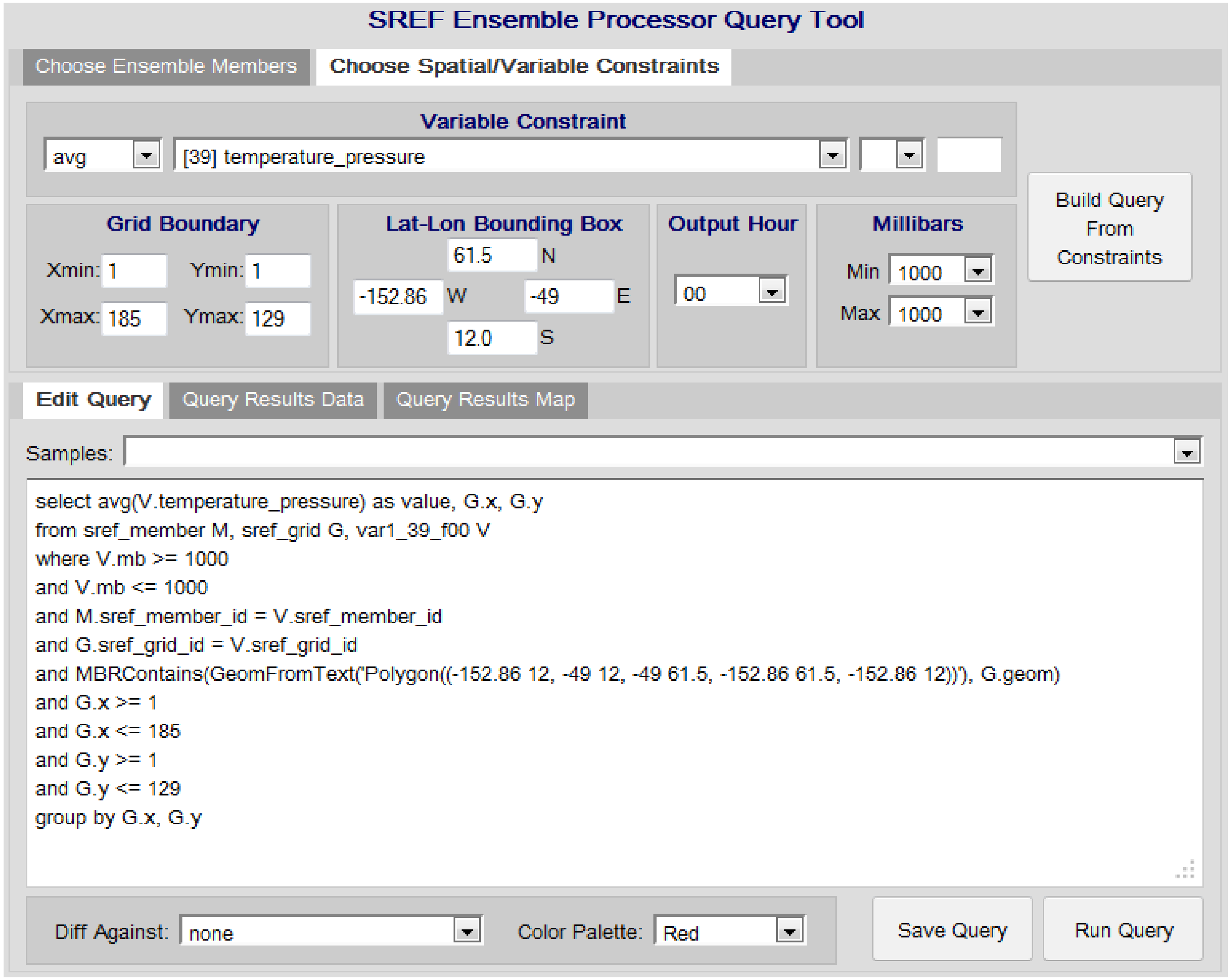
## SREF Query Tool -4

- After the user has built his query (or selected a previously saved query from the drop down list), he can:
  - Choose a color palette
  - Diff the query against the ensemble mean
  - Edit/modify the query
  - Save the query for later

Running queries can be canceled by the user (useful if he has inadvertently submitted a query to MySQL that would take hours to execute).

# SREF Query Tool -5

This query returns the ensemble mean temperature for all 21 members at 1000 mb, output hour 00, over the entire grid212 (basically CONUS).


SREF Ensemble Processor Query Tool
Choose Ensemble Members
Choose Spatial/Variable Constraints
Edit Query
Query Results Data
Query Results Map
Build Query From Constraints
Save Query
Run Query

Variable Constraint: avg | [39] temperature_pressure

Grid Boundary: Xmin: 1 Ymin: 1 Xmax: 185 Ymax: 129

Lat-Lon Bounding Box: 61.5 N, -152.86 W, -49 E, 12.0 S

Output Hour: 00

Millibars: Min 1000, Max 1000

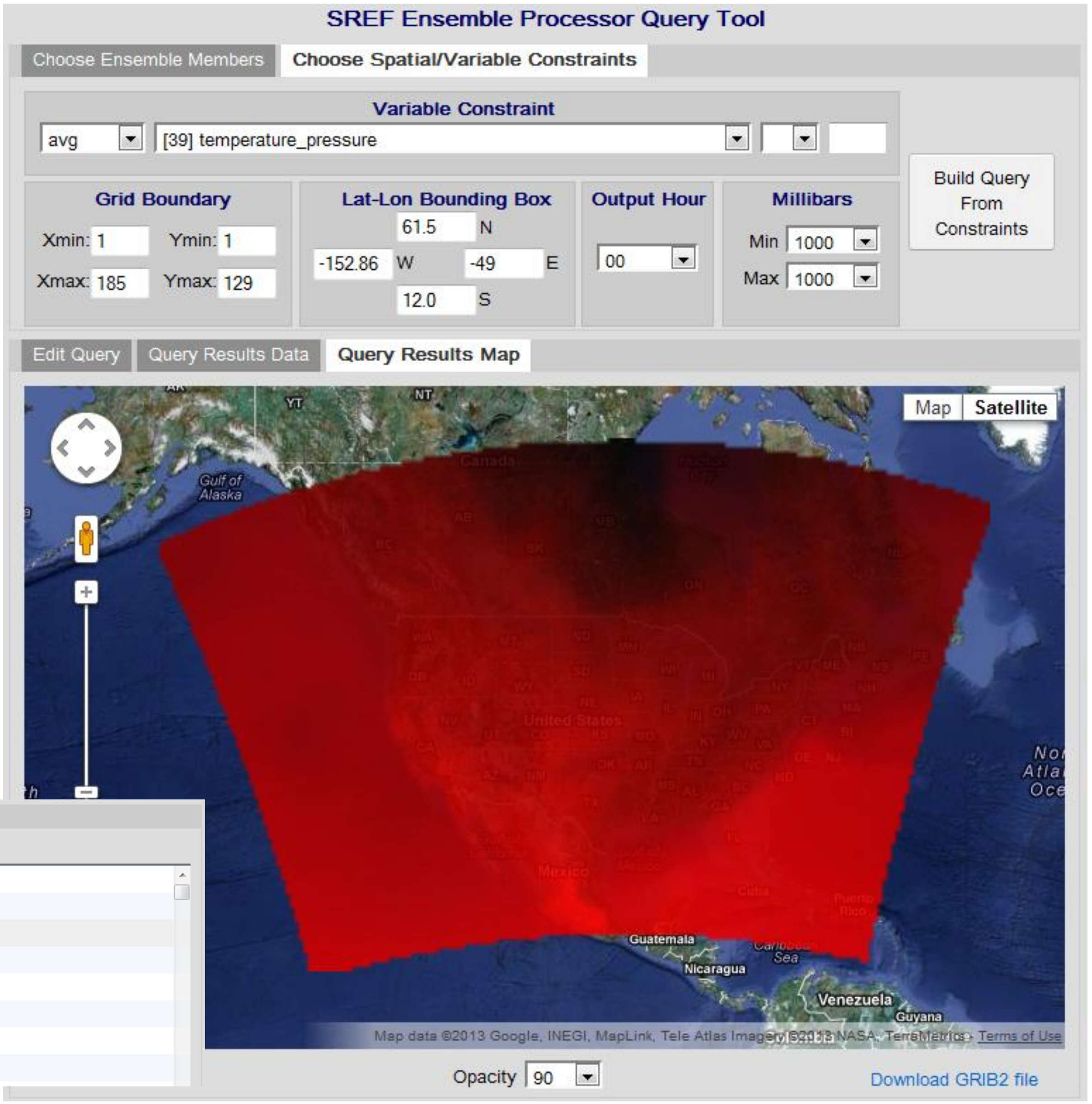
Samples:

```
select avg(V.temperature_pressure) as value, G.x, G.y
from sref_member M, sref_grid G, var1_39_f00 V
where V.mb >= 1000
and V.mb <= 1000
and M.sref_member_id = V.sref_member_id
and G.sref_grid_id = V.sref_grid_id
and MBRContains(GeomFromText('Polygon((-152.86 12, -49 12, -49 61.5, -152.86 61.5, -152.86 12))'), G.geom)
and G.x >= 1
and G.x <= 185
and G.y >= 1
and G.y <= 129
group by G.x, G.y
```

Diff Against: none Color Palette: Red

# SREF Query Tool -6

The results can be seen as
- data values
- Google Maps
- grib2 file (imported into other apps like AWIPS 2).



SREF Ensemble Processor Query Tool

Choose Ensemble Members | Choose Spatial/Variable Constraints

**Variable Constraint**: avg | [39] temperature_pressure

**Grid Boundary**: Xmin: 1, Ymin: 1, Xmax: 185, Ymax: 129

**Lat-Lon Bounding Box**: 61.5 N, -152.86 W, -49 E, 12.0 S

**Output Hour**: 00

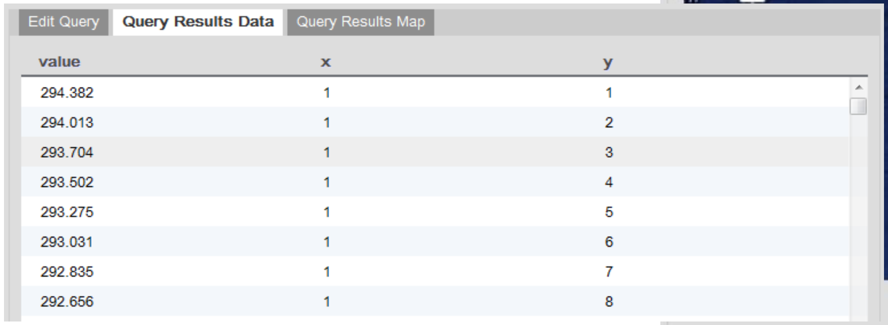
**Millibars**: Min 1000, Max 1000

Build Query From Constraints

Edit Query | Query Results Data | Query Results Map

Opacity 90

Download GRIB2 file

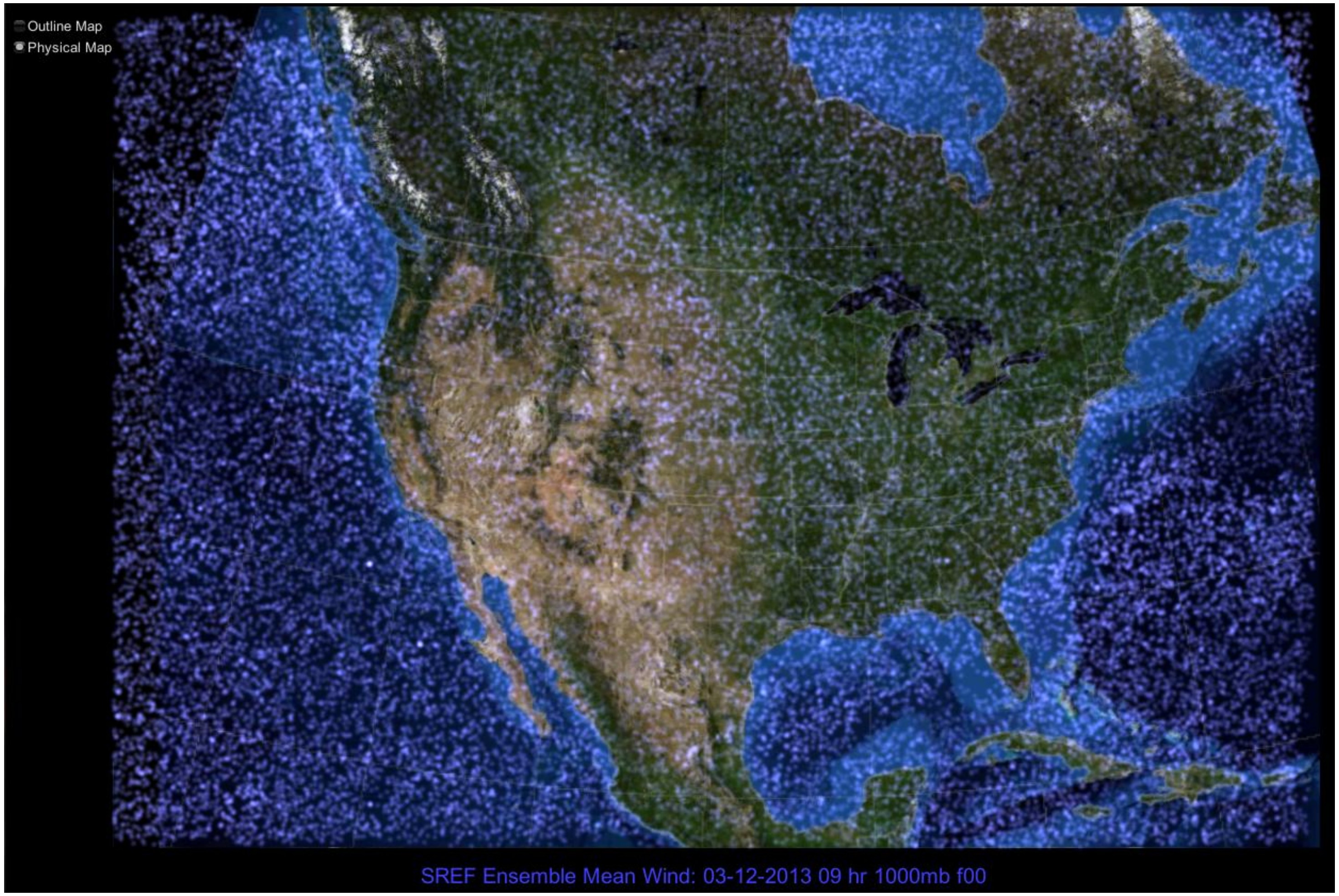


Edit Query | Query Results Data | Query Results Map

| value | x | y |
|---|---|---|
| 294.382 | 1 | 1 |
| 294.013 | 1 | 2 |
| 293.704 | 1 | 3 |
| 293.502 | 1 | 4 |
| 293.275 | 1 | 5 |
| 293.031 | 1 | 6 |
| 292.835 | 1 | 7 |
| 292.656 | 1 | 8 |

## SREF Query Tool –Wind Visualization

There is also a RESTful interface to the MySQL database. This Unity3D WindViz app creates 70,000 wind particles based on the vector field it generates from the u and v components of wind (retrieved from a RESTful web service call).



## Final Thoughts and Future Work

- Solid State drives significantly improve I/O performance.
  - On my laptop, performance went from 80 MB/s reads/writes to 480 MB/s reads/writes (for large files). Performance gains are good (but not as dramatic) when reading/writing lots of small files.
- We are moving from the 40 km SREF to the 16 km SREF (6.25 times more data)
  - We’ve ordered Intel SSD 910 800GB (2000 MB/s reads, 1000 MB/s writes) to handle this.
- There are some queries (which cannot leverage indexes) that are slower in MySQL than reading the raw grib2 files. The import program runs such a query and saves the results in MySQL for later access.

## Any Questions?

Jeff.S.Smith@noaa.gov