Real-Time Collaborative Analysis with (Almost) Pure SQL: A Case Study in Biogeochemical Oceanography

Daniel Halperin
dhalperi@cs.uw.edu

Konstantin Weitz
weitzkon@cs.uw.edu

Bill Howe
billhowe@cs.uw.edu

Francois Ribalet
ribalet@uw.edu

E. Virginia Armbrust
armbrust@uw.edu

ABSTRACT

We consider a case study using SQL-as-a-Service to support "instant analysis" of weakly structured relational data at a multi-investigator science retreat. Here, "weakly structured" means tabular, rows-and-columns datasets that share some common context, but that have limited a priori agreement on file formats, relationships, types, schemas, metadata, or semantics. In this case study, the data were acquired from hundreds of distinct locations during a multi-day oceanographic cruise using a variety of physical, biological, and chemical sensors and assays. Months after the cruise when preliminary data processing was complete, 40+ researchers from a variety of disciplines participated in a two-day "data synthesis workshop." At this workshop, two computer scientists used a web-based query-as-a-service platform called SQLShare to perform "SQL stenography": writing queries in real time to integrate data, test hypotheses, and populate visualizations in response to the scientific discussion. In this "field test" of our technology and approach, we found that it was not only feasible to support interactive science Q&A with essentially pure SQL, but that we significantly increased the value of the "face time" at the meeting: researchers from different fields were able to validate assumptions and resolve ambiguity about each others' fields. As a result, new science emerged from a meeting that was originally just a planning meeting. In this paper, we describe the details of this experiment, discuss our major findings, and lay out a new research agenda for collaborative science database services.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Scientific Databases

General Terms

Design, Experimentation, Human Factors, Management

1. INTRODUCTION

Data analysis is replacing data acquisition as the bottleneck to scientific discovery. The challenges associated with high-volume data have received significant attention [9], but the challenges related to integrating weakly structured, *high-variety data* — hundreds of datasets with hundreds of columns and no a priori agreement on format or semantics — are understudied. Even at small scales, our collaborators report that these situations require them to spend up to 90% of their time on data handling tasks that have little to do with the science [5].

We posit that the use of declarative query languages can significantly reduce the overhead of working with weakly structured relational data, allowing real-time, discussion-oriented scientific Q&A as opposed to relying on offline programming. To test this hypothesis, we have designed and deployed a web-based query-as-a-service system called SQLShare [5][1] that emphasizes a simple Upload/Query/Share workflow over heavyweight database engineering and administration tasks. Data can be uploaded to SQLShare "as is" and queried directly; a basic schema is inferred from the column headers and data types. Queries can be saved as views and shared with colleagues by exchanging urls. In prior work, we found that this approach can capture most relevant tasks and improve productivity for distributed, asynchronous collaboration [6].

In this paper, we consider whether our query-as-a-service approach can also be used to improve productivity in real-time, synchronous, face-to-face collaboration, even without assuming that the data has been integrated into some pre-engineered schema. The challenges are significant: data must be cleaned and integrated on-the-fly, and science questions must be disambiguated and encoded in SQL, also on-the-fly. If successful, this level of interactivity for scientific Q&A is not just faster, it's different. Instant results to questions arising from organic discussion changes the nature of the meeting: instead of assigning action items for investigators to complete offline when the "trail is cold," the researchers can test hypotheses and explore the implications online, during the meeting, while the ideas are fresh and everyone's perspective can be incorporated — "data-driven discussion."

We test this approach in the context of the GeoMICS project [1], a multi-institution, multi-disciplinary oceanographic collaboration between geochemists and molecular ecologists spearheaded by co-author Armbrust. The team acquired data during a research cruise in May 2012 in the northeast Pacific Ocean. The overall purpose of the cruise was two-fold. The *scientific* goal of the cruise was to study a well-defined

[1] `https://sqlshare.escience.washington.edu/`

transition zone between coastal and open-ocean waters. To do this, the PIs employed a battery of physical and optical sensors, biological and chemical assays, and sample-based measurement techniques to collect hundreds of independent physical, biological, and chemical variables at hundreds of discrete locations east of, west of, and directly inside the transition zone between shore and deep ocean. The second goal of the cruise was *collaborative*: to bring together "two largely independently operating research communities – geochemists and molecular ecologists" [1], for the first time, in order to study the same types of biogeochemical features in the ocean through the lens of very different types of data and analyses. In conversation, the scientists from different disciplines explicitly stated that they intended to "prove that we can work together" so that funding agencies would support these types of collaborations on program scale [1].

The second stage of the project was a *data synthesis workshop* hosted in February 2013, nine months after the initial cruise. One of the requirements of participation in the GeoMICS project was that all participants agree to complete sample analysis in a timely manner and share the data broadly within the group; this workshop was the first test of this mandate. Before the meeting, the team prepared preliminary datasets (primarily spreadsheets and delimited ASCII files) and centralized them via DropBox. These data were uploaded as-is into SQLShare [5] by the computer science team — each file (or sheet) became a distinct table with attribute types assigned automatically by SQLShare. At this stage, no attempt was made to integrate the data or prepare some unified schema.

Also prior to the meeting, the participants were asked to submit a set of English questions representing their science interests, an adaptation of the "20 questions" requirements-gathering methodology proposed by Jim Gray [4]. These preliminary questions helped resolve ambiguities and expose relationships in the raw data, and also served to engage with the science team and generate enthusiasm in the lead up to the meeting.

During the meeting, the group of more than 40 investigators and students came together to compare, contrast, and combine their data and insights, both within each discipline and especially across disciplines. To conduct the "field test" of our query-as-a-service technology and our schema-free approach, two computer science co-authors also attended this workshop to sit in on the meeting and act as "SQL stenographers," translating questions into SQL in real-time in response to the discussion, while working with the scientists one-on-one and in groups as appropriate to clarify the science and resolve ambiguity. Some quantitative questions were answered directly in SQLShare ("What were the top five proteins expressed at each station?") For less precise questions ("What is the relationship between temperature and salinity?"), we generated visualizations using domain-specific tools that were linked to SQLShare programmatically in the days leading up to the event.

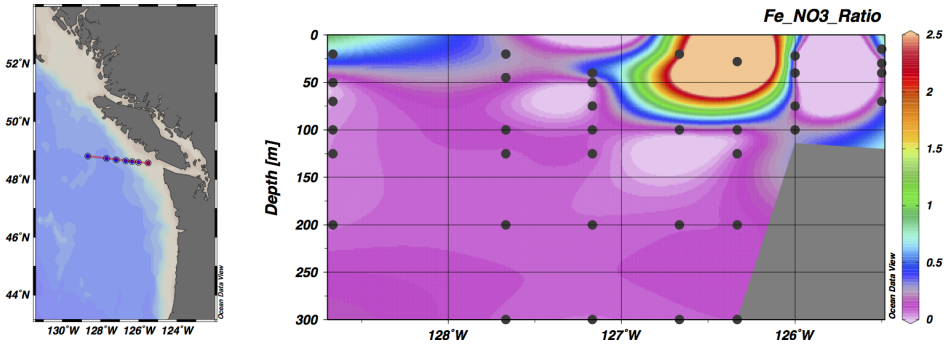
In this paper, we describe the science, the data, the queries, and our findings from this experiment, concluding that on-the-fly integration and analysis is feasible with essentially pure SQL, despite the lack of an engineered schema, the challenges of interdisciplinary communication, and the bleeding-edge science. In particular, we found the web-based query-as-a-service system to be critical: Given the SQLShare UI, we were able to refactor common queries into reusable views, organize the views using simple tagging schemes, and keep track of recent recent results during discussions. Given the SQLShare REST interfaces, we were able to avoid writing any new code to parse unusual file formats, and we were able to export data automatically to client applications preferred by the researchers. Overall, the combination of the technology and the "social protocol" used to collect data and queries by the lead PI were instrumental in the success of the experiment.

2. GEOMICS SCIENCE

In this section, we describe the scientific motivation behind the GeoMICS project.

Over the last three decades, extensive oceanographic surveys have been conducted to improve the understanding of large-scale circulation and biogeochemical cycles in the marine environment. The World Ocean Circulation Experiment, the Joint Global Ocean Flux and GEOTRACES programs, along with many other studies, demonstrate that microbial communities drive the biogeochemical cycling of the major elements (e.g., carbon, nitrogen, phosphorus, sulfur) on our planet. The results of these studies indicate that marine microbes generate and recycle about half of the organic carbon produced on Earth and carry out all nitrogen fixation, ammonia oxidation, denitrification, sulfur reduction/oxidation, and mediate the distribution and speciation of bioactive metals within the oceans. Furthermore, there is now evidence for the existence of biogeochemical oceanic provinces where large-scale chemical and physical features dictate microbial activity and the resulting elemental cycling. Recent data indicate that the oceans are undergoing rapid changes: ocean waters are warming, wind patterns are shifting, and ocean circulation is changing, together shifting turbulent mixing and delivery of nutrients from deep to surface waters. Such dramatic changes underlie an urgent need to identify the processes and quantify fluxes that control the biogeochemical cycles in the ocean. Understanding the factors that dictate province boundaries will allow predictions of how these regions may expand or contract under future ocean conditions.

To explore these questions, two largely independent operating research communities—geochemists and molecular ecologists—conducted a cruise to the northeast Pacific Ocean to sample the province boundary between offshore High-Nitrate, Low-Chlorophyll and coastal waters. The aim was to examine the interactions between changes in microbial diversity, community functions, and chemical features. The transition zone between these two oceanographic provinces is characterized by a strong gradient in biogeochemical properties and high biological activity. The oceanographers collected an unprecedented suite of biological samples (metagenomes, metatranscriptomes, metaproteomes and metabolomics of viruses, bacteria and phytoplankton) and chemical measurements (nutrients and dissolved and suspended particulate iron, copper, zinc, manganese, cobalt, nickel and cadmium) from surface to seafloor. Together, these samples will help elucidate the interactions between changes in microbial diversity, community functions, and chemical features at relevant spatial resolution.



1: An example visualization product from the workshop. The map at left shows the locations of the seven measurement stations visited during the cruise, labeled P1 through P8 (excluding P7) from east to west. The plot at right shows the ratio of iron (Fe) concentration to nitrate (NO_3) concentrations throughout cruise area to a depth of 300 m. To generate this figure, we used SQLShare to join three disparate datasets—metal concentrations; nutrient concentrations; and an external bathymetry dataset—and then compute the Fe/NO_3 ratio using NULL-and-0-aware division. We then used our Ocean Data View adaptor webapp to download the derived dataset, imported it into ODV, and generated this visualization. In the plot at right, each black dot shows locations where the ratio was defined, and the colors are automatically interpolated by ODV.

The data collected from this cruise, and the collaborative process by which the data will be analyzed, poses a challenge for conventional database technology. These highly interdisciplinary, highly collaborative projects are characterized by extremely heterogeneous data sources, diverse user backgrounds and skillsets, and a need for real-time collaborative analysis. This paper represents an initial case study testing a query-as-a-service platform.

3. METHODOLOGY

In preparation for and during the data synthesis workshop planned by the researchers, the team applied the following methodology.

A minimal standardized data template was prepared and mailed to all GeoMICS participants. Each row of the template includes six attributes (Event, Latitude, Longitude, Station, Depth, Source) that together represent a specific sampling event, the location, the depth below the surface in meters, and the instrument used. These semantics were not enforced or validated; as a result, many datasets required some transformation before they could be integrated.

After the distributing the template, the Lead PI sent an email asking all GeoMICS participants to upload their data to DropBox. Just before the workshop, roughly 80% of participants had responded and provided their data. Once the data appeared in DropBox or were sent by email, we uploaded the data sets to SQLShare, which parsed the basic structure of the data and inferred column types. While most data was ingested into SQLShare automatically, some datasets exposed bugs in SQLShare's parsing capabilities: illegal column names containing bracket characters needed to be replaced offline, for example.

Once the data was uploaded, additional formatting and cleaning steps were performed in SQL directly. For example, numeric values were suffixed with their units, making SQLShare interpret the value as a string. Interestingly, once the researchers saw their data in SQLShare and recognized the ability to combine data from different sources, we were swamped with emails asking for help "attaching" data. These questions exposed some misunderstandings — some researchers believed that re-uploading data was the only way to combine two datasets. The ability to write queries to accomplish the same task was not immediately obvious.

The PI also solicited representative questions capturing science goals; this process was modeled after Jim Gray's "20 questions" approach [4]. We described how they were evaluated in Section 6.

To facilitate analysis using tools familiar to the researchers, we also built client applications against SQLShare's REST API (which took approximately one day — a very low level of effort.) These tools are described in more detail in Section 4.

The meeting was organized as a plenary session on the first day, then two breakout sessions on the second day (one by discipline, and another intentionally interdisciplinary). On the first day of the workshop, we gave an introductory presentation to the group on SQLShare, but offered no additional training in either SQLShare or SQL. We relied entirely on hands-on support during the breakout sessions (Figure 2). The results of these sessions and a discussion of lessons learned are provided in Section 7.

4. THE TECHNOLOGY

4.1 SQLShare

At the University of Washington eScience Institute, we are developing a new "delivery vector" for relational database technology called SQLShare, and studying how it can be used to satisfy both scientific workflow requirements and ad hoc interactive analysis. SQLShare is a web-based query-as-a-service system that eliminates the initial setup costs associated with conventional database projects and instead emphasizes a simple Upload/Query/Share protocol: users upload their table-structured files through a browser (or programmatically via an API) and can immediately query them using SQL — no schema design, no preprocessing, no database administrators. SQLShare users derive new datasets by writing and saving SQL queries. Each derived dataset

2: Questions that came up during breakout sessions were translated into SQL queries in real time, without requiring any up-front schema design or application building efforts. Here, the lead author presents the iron/nitrate ratio plot (Figure 1) to researchers from a variety of backgrounds.

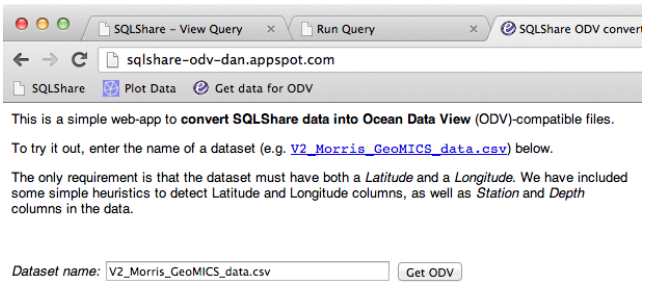
is managed as a view in the underlying database: a named, permanently stored query. Each dataset is also equipped with descriptive metadata. Everything in SQLShare is accomplished by writing and sharing views: Users can clean data, assess quality, standardize units, integrate data from multiple sources, attach metadata, protect sensitive data, and publish results.

SQLShare has three components, all of which are cloud-hosted: a web-based UI, a REST web service, and a database backend. The UI is a Django Python application (a web framework similar to Ruby on Rails), and hosted on Amazon Web Services. The UI communicates with the backend exclusively through REST calls, ensuring that all client tools have full access to all features. The web service is implemented on Microsoft Azure as (one or more) Web Roles. The database is implemented using Microsoft's SQL Azure system, which is very similar to Microsoft's SQL Server platform.

Version 1 of SQLShare was completed in 2010 and has been in use by scientists at UW and elsewhere since that time. To complement and extend its functionality for this experiment, we also produced several custom tools which interact with SQLShare via the REST API. These tools were designed to add domain-specific interfaces for oceanography to SQLShare, to provide online, ad hoc, scriptable visualizations, and for assistance in writing SQL statements to execute data cleaning and integration tasks. Some of these tools served as early prototypes for features that might be added as core SQLShare functionality in a later version.

4.2 SQL generators for common patterns

A key advantage of using SQL for science is that relational algebra can express core data cleaning, integration, and analysis operations natively: cleaning is usually transformation with `SELECT` statements, and integrating data from different datasets is usually a JOIN with the appropriate condition. Unfortunately, actually writing these SQL expressions can be complicated, time-intensive, and error-prone. To aide in this task, we developed scripts to programmatically generate SQL statements using common idioms.



3: The SQLShare Ocean Data View adaptor runs as a free Google App Engine web application. It automatically reformats datasets from SQLShare so that they can be automatically imported into ODV, a domain-specific tool for data visualization, analysis, and presentation.

One example would be to create a view that contains all columns from a source table with one column renamed: there is no standard SQL construct to do this in a concise manner — each and every column must be explicitly mentioned in the `SELECT` clause. But given simple syntactic sugar to expand `SELECT * FROM TABLE` into `SELECT col1, col2, col3 FROM TABLE` a user can easily modify this to perform the desired operation by changing `col3` to `col3 AS col4`.

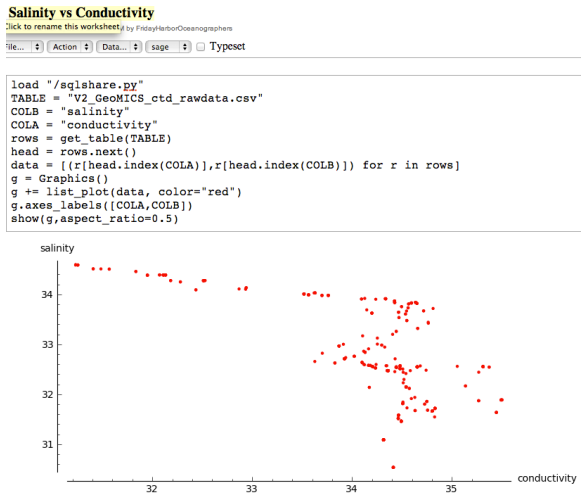
A second example is a global find and replace. Null values are frequently represented by some domain- or researcher-specific sentinel value in the data: oceanographers prefer -99, for example. Replacing nulls in a single column involves a simple `CASE` expression, but replacing values in all columns involves an enormous amount of typing. Automatically generating the appropriate queries to replace values in every column in which they appear reduced effort.

4.3 Ocean Data View adaptor

Oceanographers commonly visualize data in a domain-specific tool called Ocean Data View (ODV).[2] ODV accepts plain rows-and-columns CSV or text files as input, however the process of importing an arbitrary file is time-consuming and manually intensive because the user must associate each column with a known data type. Any errors made during this import process typically cause users to abandon their progress and start over from the beginning—indeed, we saw this happen at least twice during the workshop when users attempted to import datasets not produced by our adaptor.

This manually-intensive process can be avoided if the input file has a shape that ODV recognizes. In particular, the order and names of columns are important. The name of the cruise, the station of the measurement, latitude, longitude, and depth must be located at the beginning of the table. And ODV will not detect that a field called 'Latitude..Decimal.deg.' (a sanitized form of 'Latitude (Decimal deg)' found in a dataset produced by R) is a latitude and thus the automatic import will fail. Using the SQLShare REST interface, however, we can automatically inspect the columns of a dataset, determine whether it is compatible with ODV, and then programmatically construct the SQL statement to permute and rename columns so that the resulting dataset can be automatically imported into ODV. We packaged this

[2] `http://odv.awi.de/`



4: The Sage Notebook runs in the Amazon Cloud. In this example, we use the SQLShare REST API to download tables, and simple Python commands to plot the relationship between two columns.

logic into a Python web application, which we then deployed on a free Google App Engine instance. This tool enabled anyone at the workshop to visit the application, enter the name of an original or derived dataset, and download a transformed version of that dataset ready for automatic import into ODV. This tool enabled users, some of whom may have never logged into SQLShare nor seen SQL code, to easily import, visualize, and analyze data without any input or assistance from the computer scientists at the workshop.

SQLShare's REST interface made our tool development particularly easy: an author who had never used REST before was able to develop both of these tools from concept to production in about twelve hours.

4.4 Sage notebook adaptor

We extended SQLShare with support for basic visualizations using Sage. Sage is an open source bundle of mathematics libraries and programs, intended to provide many features of well known programs such as R, Mathematica and Matlab. We set up a Sage Notebook Server on an Amazon Web Services instance and provided a function get_table that uses SQLShare's REST API to download a table and make it directly accessible in the notebook. The entire design, development, and deployment process was completed in less than an hour.

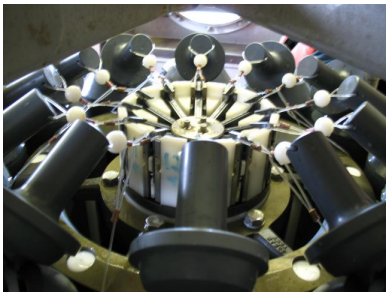
5. THE DATA

In this section, we describe the measurements made during the cruise and then discuss the characteristics of the processed datasets that were available for the data synthesis workshop.

5.1 Measurements

Three categories of measurements were collected:

Underway data were collected by analyzing the seawater flowing continuously through a sensing apparatus mounted underneath the vessel at a fixed depth of three meters below the surface. This platform was equipped with a continuous



5: The top of the rosette of Niskin bottles, a water sampling apparatus. These caps snap shut, trapping the water inside as the sensor package descends through the water column. Each bottle can be closed electronically at a different depth, and captures 10 L of water – enough for a variety of chemical and biological assays.

flow-throw thermosalinograph (measuring salinity and temperature, as well as conductivity) and a SeaFlow [11] device that uses flow cytometry to count and classify the organisms in the water. Other data include latitude and longitude from the ship's navigation system, and environmental factors such as air temperature, humidity, and barometric pressure.

Sample data were gathered when the ship was anchored on station, using one of three techniques:

- In a *cast*, a rosette of up to twenty-four empty 10 L bottles (Figure 5) were lowered, while attached instruments continuously sampled data such as temperature, salinity, fluorescence, and nitrate concentration. Based on this depth profile, where measurements were taken at other stations, and the intended use of the water samples, the scientists then chose at which depths to collect water samples and how many bottles to close. The rosette was then raised to the surface, while each bottle shut at its chosen depth to capture a snapshot of the water at that specific time and location.
- A *surface pump* continuously pumped water from 5 m below the surface up to the deck of the ship. Some of the water samples were filtered through various size filters to collect particles and organisms of particular sizes, and in other cases, particle-free seawater was gathered.
- A special *McLane pump* was configured with filters of chosen sizes, lowered to a specified depth, and then filtered water continuously until either the exiting water pressure dropped (i.e., the filter clogged because it had collected enough samples) or a fixed time period expired. The McLane pump emulates the surface pump operation, but at depth.

Some samples were processed in a lab space on ship, while others—typically the filtered organic samples for genomics, proteomics, or transcriptomics (collectively referred to as "-omics")—were frozen, catalogued, and analyzed in the individual scientists' lab facilities. In addition to the features described above, concentrations of trace metals (copper, iron, zinc, cadmium, manganese, nickel and cobalt) and inorganic nutrients (nitrate, nitrite, ammonium, phosphate, silicic acid, dissolved inorganic carbon, and biogenic silica) were measured. For all sample types, multiple samples were often collected at each station for the purposes of scaling up the number of samples or repeatability of the measurements.

Finally, **cast data** were collected from the instruments attached to the rosettes lowered during casts. These data give a view through the water column of physical and biological properties. Like the sample data, these data are collected at similar times and the same location, but they capture the entire water column rather than a few discrete depths.

5.2 Ingested Data

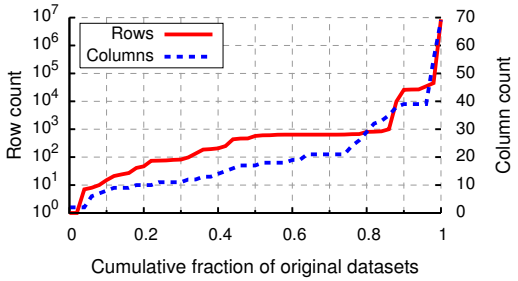
In total, 49 original datasets were available and uploaded to SQLShare for use at the data synthesis workshop. These data represent preliminary offline processing of most of the physical and chemical variables. In contrast, only a few of the biological datasets were prepared in time for the workshop, in part because of the labor- and CPU resource-intensive processing steps to generate these data. Additionally, the integration-heavy theme of this workshop uncovered another process issue with varied biological data, discussed below in Section 7.4. There was some resistance to sharing data before it was fully quality controlled.

Organization. The GeoMICS data synthesis workshop was distinctive in the sense that the cruise had been planned with collaboration in mind[3]. To this end, every researcher was instructed to associate each measurement with a shared key. In particular, the following combination of four factors can uniquely identify each sample: which *Station* the ship was at, which *Instrument* collected the data, from what *Depth* the sample was taken, and during which *Event*—a cast, a position of the surface pump, a lowering of the McLane pump, etc.—the sample was gathered. The science team promoted the concept of a unique label formed by concatenating the four attributes participating as a key, but we typically ignored this surrogate key (which violates first normal form) and referred to the composite key directly in our queries. In addition to the five columns described, the template also included a Latitude and Longitude column containing the locations of the Stations. Though redundant, these columns were intended to aid scientists in linking their data with the template and as a sanity check in case the human-generated Station label on a dataset was incorrect.

To collect data for the workshop, an Excel spreadsheet containing all valid four-column-keys was distributed to the researchers, who appended the columns containing their data and then uploaded the datasets to a shared repository setup by the Lead PI. Some datasets used only a subset of the key: the -omics data were collected only at the surface using the surface pumps and hence included only the Station number, and the underway data used Latitude and Longitude as keys because they were collected at all locations, not just on Station.

Example. Table 1 shows a partial subset of a trace metal dataset uploaded to SQLShare. The left four columns give the composite key identifying the cast, station, depth and instrument at which the water analyzed to obtain these values was sampled. The right four columns show the concentrations of iron (Fe) and copper (Cu), as well as error bars given by their standard deviations. This snapshot is only part of the entire table; the original dataset has 15 columns, including some redundant key information (Latitude, Longitude, and Label) and the concentrations of zinc and manganese, and 74 rows, corresponding to additional depths and stations. During science discussions at the workshop, this dataset was joined with other datasets to generate the visualizations of both Figure 1 and Figure 9.

[3]This project was unusual: While nearly all cruises host the experiments of many independent researchers, these different measurements are practically never integrated!



6: The number of rows and columns in the 49 original datasets. Dataset size covers seven orders of magnitude from 1 to 10 million rows. The number of columns ranges from 2 to 69: most datasets measure few variables, but several measure 30 or more.

Characteristics. Figure 6 shows the distribution of the number of columns and rows in the 49 datasets collected. The number of rows indicate how many measurements each dataset contains, ranging from 1 row to 10 million rows—a spread of seven orders of magnitude. The input data contain both measurements and external datasets, and each can be large or small. At the large end, both continuously-sampled underway data and the biological protein or gene sequence data extract from stored samples lead to datasets with many rows, while at the small end some data is taken only once a few times throughout the cruise, such as instrument calibration. Similarly, some external databases that we imported are very small, such as the bathymetry data representing the depth of each station. The largest dataset was a BLAST function map, which maps database-specific gene IDs to human-generated descriptions of their biological functions.

The number of columns indicates the number of variables measured in each dataset. Many of these datasets only measure a few quantities: over 75% of the input datasets have 20 or fewer columns, but several datasets cover dozens of variables and have up to 69 columns. Datasets with fewer columns are typically external datasets that do not fit the template. For instance, the bathymetry dataset above has only two columns: the Station and the corresponding Depth of the ocean floor.

5.3 Dataset idiosyncrasies

We uncovered a number of idiosyncrasies in the source data provided by the scientists. Here are a few examples:

Template misuse: In some datasets, the Latitude and Longitude columns were switched. We suspect this was caused by the creator copying and pasting columns and column headers from the template into a pre-existing spreadsheet, rather than copying the data into the template. In another dataset, the leading 'P' was omitted from Station names, e.g., P8. We were able to repair both these errors by modifying the wrapper views in SQLShare, in the former case by expanding `SELECT * FROM table` to `SELECT Event, Latitude AS Longitude,`

Event	Station	Depth (m)	Source	Total Fe (nM)	Stdev Fe	Total Cu (nM)	Stdev Cu
1004	P8	20	Niskin	0.265	0.012	1.389	0.036
1004	P8	33	Niskin	0.166	0.005	1.052	0.01
1004	P8	50	Niskin	0.164	0.027	1.089	0.007

1: A sample of the first few rows and columns of a dataset containing the measured trace metal concentrations. The left four columns give the composite key identifying the cast, station, depth, and instrument at which the water used to measure these values was sampled. The right four columns show the measured concentrations of iron (Fe) and copper (Cu), as well as standard deviations on these values. These data, provided by Jagruti Vedamati, were used in generating both Figure 1 and Figure 9 live at the workshop.

`Longitude AS Latitude, Station, ...` to swap the column names. In the latter case, SQLShare had automatically detected that Station contained only integers, so we had to modify the view to cast it to a VARCHAR and prepend a 'P'. In both cases, we caught these problems when debugging queries in real time, e.g., "Why are there no results for this join?", and were able to rapidly correct them directly in SQL.

Embedded tables: Some uploaded spreadsheets contained multiple "tables." In a table of protein sequence and BLAST data, the first 50 or so columns contained data about the proteins found when sequencing the data and their annotations. To the right of these columns, the scientist had added aggregate results such as "Given the column identifying the most specific taxonomic group (e.g., family, genus, species) this protein is found in, how many copies of the taxa were found?". We were able to express these derived tables as views over the raw data, and then we modified the SQLShare wrapper to remove those excess columns from the base dataset. This change highlighted another advantage of SQLShare: the tables in Excel were not stored as programs, hence the method by which they were computed was not obvious, and only represented a snapshot of the source data. In contrast, because SQLShare stores queries as views, it both documents the provenance of derived data and automatically stays updated when the base datasets change. When shown the SQLShare approach, the scientist recognized the value of the approach and was "excited to see more".

Data in file names: Data was frequently embedded in the names of uploaded files. For example, recall that cast data from thermosalinographs (TSGs) and other instruments are collected continuously during every lowering and raising of a rosette. The natural procedure in this case was for scientists to save one file for every individual cast (identified by the Event column in our composite key). When uploaded into SQLShare, these files were named 'event1001.cnv', 'event1004.cnv', etc. (CNV is a proprietary format used by Sea-Bird Electronics in thermosalinographs and other sensors). We were able to combine these datasets using a UNION ALL statement of the form `SELECT *,1001 AS event from [event1001.cnv] UNION ALL ...` to form a single unified cast data table.

Excessive processing: In several cases, uploaded datasets were overly pre-processed. For the protein dataset referred above, the table was actually a composite of many datasets joined together. The initial few columns contained a protein group identifier, the number of unique peptides on that protein, and the count of how many times it was found in the sample. The next 6 columns contained the top 3 functional annotations (i.e., human descriptions of what scientists believe that protein does) associated with that protein and the relative abundance of the selected function among all associated functions. When asked why he chose to do it this way, the scientist admitted that 3 functional annotations was an arbitrary choice, but that he needed to make some choice to be able to fit the data into a usable spreadsheet form. In this case, we wished we had the raw data: we could have stored the initial columns describing the sequenced protein counts separately from the functional annotation database, and we could create one or more views to produce the top 3 annotations, the top 5 annotations, or even vary the number of annotations based on a per-protein significance criterion.

6. PREPARING FOR THE WORKSHOP

In advance of the workshop, the Lead PI solicited English questions from the project participants. One intention was to get the scientists into a preparatory mindset for collaboration and integration at the workshop. The second goal was to familiarize the computer science team with both the uploaded datasets and the domain terminology we would encounter at the meeting. This enabled us to better assimilate into the science discussions and more easily express queries in SQL in real time.

The responses were collected in a shared Google Doc. Before the workshop, there were 6 queries; after the workshop we had received more than 30. Here are a few examples, roughly in increasing complexity:

1. "What is the relation between Syn. (bacteria, virus, Micromonas) concentration and temperature (salinity)?"
2. "When the iron concentration is below X nm, how many iron related proteins (based on annotations) are detected?"
3. "What are top 5 highest concentration organisms based on cell number (based on proteomes, based on genome) at a given depth or site?"
4. "Can we use a subset of well-understood phenomena to 'ground truth' the GeoMICS approach, documenting (hopefully) the expected linkages between genes, transcripts, and proteins, on the one hand, and processes and stocks, on the other? Examples: photosynthesis rate, limiting nutrient, plankton composition, growth rate, etc."

In this section, we describe the process by which we answered these English questions using SQL.

Phase I: Translating existing analyses. The week before the meeting, the science and computer science teams at UW met face-to-face and electronically to review the organization of the data, examine two datasets, and discuss the (intra-dataset) analyses the scientists had already been performed, such as the protein taxa counts found in the embedded tables

described in Section 5. We were easily able to translate existing analyses into SQL, as these queries mapped directly into group-by and aggregate statements. This phase also provided an opportunity for the computer science team, who had used SQLShare only sparingly prior to this meeting, to familiarize themselves with the system.

Phase II: Answering novel queries. Next, the computer science team attempted to answer the 6 sample questions using the 30 or so datasets available (with more coming in). At first, this process was extremely difficult: datasets were often named for the *category* of data (e.g., 'nutrients'), while questions referred to *features* (e.g., 'phosphate', 'nitrate', or 'ammonium'). The names of columns storing a feature and the English words describing them did not align (e.g., 'Tot.Fe.nM' vs. 'iron concentration'). Consequently, we began Phase II by opening each dataset in SQLShare, reading its column names and looking at a few sample rows, and perhaps performing a web search to look up certain terms. Next, it was often unclear by what features datasets should be joined. Two different datasets will rarely contain measurements using equal composite keys: this can be true only if they were taken by the same Instrument during the same Event, i.e., using water samples from the same bottle on the same cast. Is it appropriate to join measurements from different casts? Does Depth have to match exactly, or can it be close (and what does 'close' mean)? Many -omics datasets omitted Depth information entirely; we later learned that these datasets were collected at 5 m depth. As we found when discussing with science colleagues, in practice, scientists integrate data using Station and Depth, comparing data from different instruments and from different casts—but at the same physical location.

Concrete Example: Query 2. Consider Query 2 above, which seeks to link iron concentrations with iron-related proteins. The data containing the iron data was easy to find using keyword search for 'Fe'—[V2_GEOMICS_Fe-Cu-Mn-Zn-Vedamati.csv]—and we gained confidence in this guess when the units used in the English query, 'nm', nearly matched the apparent units of the column header, 'Tot.Fe.nM'.

Next we needed to find and count iron-related proteins. The protein dataset was one of those we analyzed in Phase I, so we already knew which dataset to use. However, the functional annotations in this dataset were human-generated text such as "ribosomal protein L5"—How do we determine which of these are related to iron?

We used SQLShare's interactive querying functionality to develop this query. We started by filtering the table using the obvious `WHERE` clause, `Function_1 LIKE '%iron%'`, and found 10 matches. Next, we added `OR Function_1 LIKE '%Fe%'`. This query returned 95 matches, including false matches such as "Polyribonucleotide nucleotidyltransferase". Because of the interactive nature of writing queries in SQLShare, we spotted the problem instantly: the LIKE operator is case-insensitive in MS-SQL, so the common trans**fe**rase-related proteins would become false positives. We then replaced this second clause with `OR Function_1 LIKE '%Fe[^a-z]%'` to catch instances of the literal 'Fe'. Though this formulation is imprecise, we observed that it did not result in false positives. Finally, we added a third clause to catch instances of "%ferr%", as in "ferrous" or "ferrodoxin", which expanded the result to 19 proteins. By combining across all three functional annotations, the result expanded to 28 rows. Though this may seem small, this preliminary database of annotated proteins contains only 3130 rows, so these represent about 1% of all identified proteins.

```
WITH SurfaceMetals AS
       (SELECT Station,MIN([Depth..m.]) AS MinDepth
        FROM [V2_GEOMICS_Fe-Cu-Mn-Zn-Vedamati.csv]
        GROUP BY Station)
SELECT iron.Station
     , iron.[Tot.Fe.nM.]
     , SUM(protein.spectra_counts) AS ProteinSpectra
     , protein.[Depth..m.] AS [SurfacePumpDepth..m.]
     , iron.[Depth..m.] AS [NiskinDepth..m.]
FROM [V2_GEOMICS_Fe-Cu-Mn-Zn-Vedamati.csv] iron,
     [Iron-related_Proteins] protein,
     SurfaceMetals
WHERE protein.Station = iron.Station
  AND iron.[Depth..m.] = SurfaceMetals.MinDepth
  AND SurfaceMetals.Station = iron.Station
GROUP BY iron.Station
       , iron.[Tot.Fe.nM.]
       , iron.[Depth..m.]
       , protein.[Depth..m.]
```

7: The SQL to answer Query 2 finds the relationship between biological expression of iron-related proteins and the actual concentration of trace iron in the water.

Having developed a suitable query to identify iron related proteins, we saved this view as a derived dataset in SQLShare, [Iron-related_proteins]. This isolates the logic to identify iron related proteins from the rest of Query 2, labels the saved logic as a dataset with a descriptive name that can be found by keyword search, and enables this intermediate result to be reused in answering other questions. Additionally, should we need to amend the clauses in the filter for more accurate protein identification, this view can be edited later and the derived views will automatically reflect the changes.

Finally, we needed to join the iron related protein data with the iron concentration data. This was a simple join between these two datasets based on depth. However, in this case there were actually no depths that matched: the -omics protein data was collected 5 m deep, while the shallowest iron measurements came from 10 m down. In this case, we simply selected the shallowest iron measurement at each station. The final query we used is shown in Figure 7. This view defines a dataset that contains, for each station, the iron concentration and the total count of iron-related proteins. We also include the depth of the source of each dataset in case these are useful in later analysis. Note that while this dataset does not directly answer Query 2, for any concentration value X a trivial `SELECT` statement will compute the answer.

Summary and Lessons Learned. This section has shown how we were able to begin answering English science questions in SQL. As exemplified by the iron example, the techniques used were manifold but were easily mapped into SQL. Ultimately, we were successful in expressing the science questions in SQLShare for the queries available ahead of the workshop. Some queries, like Query 4, appear too abstract to be answered with SQL.[4] However, when talking with sci-

[4] At the workshop, one science participant proposed the query `SELECT paper FROM data`!

entists we were often, though not always, able to "compile down" these abstract queries into concrete questions.

One takeaway from Phase II was that the initial reading of the sample queries and exploration of the datasets would have been much faster if done in a tight feedback loop with the scientists that conducted the experiments. Conversely, we found that the manual process of looking through each dataset and identifying what it measured was useful, because we were then able to answer questions on-the-fly at the workshop without too much assistance from the scientists. This limited the disruption to their conversations. In practice, a balance should be struck: the computer scientists need to know enough about the data to work quickly and independently, but the ability to ask scientists questions about the nature of the datasets with short turnaround for answers could clear up issues in seconds that would take hours to work out alone.

After completing Phase II, we did modify the Google Doc collecting English science questions to request that scientists adding questions also indicate which datasets they thought contained relevant data. This change reduced the initial grokking time for each question. We also proposed a new feature for SQLShare: expanding the search facility to search over column names as well as dataset names, tags, and descriptions. This ability would also have reduced the time to map queries to data in Phase II. This is only one of many new SQLShare features to arise from this workshop; we describe more in Section 7.

7. AT THE DATA SYNTHESIS WORKSHOP

At the workshop, more than 40 investigators and students came together to review their data as a group, develop questions for further investigation, and draw some interesting conclusions from the assembled data. Much of the discussion revolved around what was different in different parts of the ocean—either east and west across stations, or vertically between different depths through the water column.

Our experiences at the meeting repeatedly underscored the value of interactivity for these types of integrative, collaborative summits. In this section, we summarize what happened at the workshop, present illustrative anecdotes showing the usefulness of our tools and approach and the transformative nature of interactivity. We conclude with a list of lessons learned.

7.1 Workshop schedule

The 2.5-day workshop began about 4 pm the first afternoon. Armbrust, the Lead PI, began with a short welcome and introduction to all the collaborators, then four scientists, representing four types of data, gave short talks with aggregate summary's of individual participants' initial findings in that area. The goal of these talks was to share concisely with the group the scope of data available to be analyzed at the workshop. The lead author then gave a brief overview of the computer scientists' role and goals, and demonstrated the SQLShare, ODV adaptor, and Sage adaptor tools. He used the SQL for the example queries described in Section 6 to illustrate the types of analysis available for the meeting.

The rest of the workshop consisted of three half-day breakout sessions in which four groups of scientists spread across two rooms. One computer scientist would roam each room, observing the discussion and looking for opportunities to help answer questions. On the morning of the second day, four groups doing science with similar types of data—eukaryotic molecular organisms, prokaryotic molecular organisms, underway/metals, and organics—met together to determine the key findings of their data. In these groups, we were often asked to answer questions such as

5. "What is the correlation between each of the following metals and phosphate? Fe, Cu, Mn and Zn."
6. "What is the relation between Zinc and Cobalt?"
7. "Can you plot Virus Count vs. Salinity?"

During the afternoon session of the second day, the four groups above were sliced horizontally and mixed and matched, with the goal of comparing data across domains, including taking the *within-group* findings of the morning and trying to correlate them *across groups*. These queries were more abstract and required combining more datasets, for example:

8. "Combine all data sets, so that we can make sense of them using ODV. Let us start by combining GDGT [Glycerol Dialkyl Glycerol Tetraether lipids], Carbon, and O2."

On the third day, the scientists split into groups to discuss logistics for the future of the GeoMICS collaboration. This included: developing a schedule of how remaining samples should be split up and analyzed; specifying holes in the collected data to be improved in the next cruise; and choosing topics for a "project report" paper to be written this summer. During this time, the computer scientists mainly worked one-on-one with a few scientists continuing the analyses of the previous day and also beginning work to analyze the key findings of this project towards the group paper.

7.2 Observations of oceanographers

We made the following observations while helping integrate and analyze data and during side discussions at the workshop.

Available technology Oceanography is an empirical science: almost all papers are based on the interpretation of data. As computer staff are very expensive, often even more costly than a full-time researcher, many of the groups have either no computer staff, or one person to help an entire department. Many PIs had used IT staff, or volunteer undergraduate help, to set up scripts, databases, or other processing pipelines in the past, but those tools were abandoned when the staff left or the students graduated. *These anecdotes underscore the value of our query-as-a-service approach.*

Visualization It is already impossible to grasp a dataset of 1,000 rows, something nobody in the the database community would refer to as big data. Consequently, data visualization is a key aspect of the way oceanographers get insight into the data generated by their experiments. Many at the workshop often used our Sage webapp to quickly visualize 2-variable datasets, used ODV (via our web adaptor) to analyze multivariate relations, or downloaded an integrated dataset and visualized it in R. For complex analyses, the choice of technology was largely based on familiarity—though *the ODV*

adaptor greatly speeded-up context switching time—but across users *the Sage workbook saw a lot of activity because of its low-overhead, fast-response-time interface.*

Data Manipulation and Integration Surprisingly, we did not see people using R to actually manipulate the data or combine it with other data sets (using for example R's merge feature) on the fly. This surprised us because this task is relevant not only in cross-researcher integration tasks like at this workshop, but also to compare the data with external data sources, like gene databases or satellite pictures. The data format of these sources may be relational, but may also come in the form of an image or XML. Without our help, the main means by which researchers compared their data with other data, was by looking at pre-generated data visualizations (for example in a powerpoint presentation) of the other data, rather than merging the raw data. For these reasons, it appeared that *without our help most cross data set questions could not have been be answered on the fly, even though they were resolvable using the data at hand.*

Hidden Semantics One reason it is challenging to integrate data from multiple sources is that the data's semantics are not sufficiently captured. Many details are only available in a researcher's head, and data transformations, both by hand and with scripts, are often not properly documented, such that the detailed provenance of data is lost. When asked, *many researchers agreed that they would not be able to recreate a paper from their raw data two years later.* This makes it impossible to compare or combine results from publication, for example to get higher statistical significance.

SQLShare helps resolve these issues by making it easy for researchers to tag and add metadata to datasets, including calibration parameters. In our own exploration of the data, we used SQLShare to compute some aggregates such as "At what stations and depths were the most water samples collected using Niskin bottles?", thereby gaining insight as to how much and which types data was collected during the cruise. One would normally have to develop a custom R or Python script to access this information, but *with the experimental metadata stored in a database, this information is "only a query away".* Additionally, *SQLShare's view-oriented strategy documents data transformations* so that they can be scrutinized after the fact. Repeating a scientists' analysis or recomputing it with updated data requires only re-executing the `SELECT * FROM view` query.

7.3 The role of interactivity

The availability of our interactive tools changed the nature of the meeting from "planning" to "doing."

Interactive hypothesis generation and testing For one of the first queries, a PI and her student requested a plot of total virus counts as a function of salinity. With the scientists directing us to the relevant datasets, we computed the join in SQLShare and generated a graph in Sage, which showed a strong linear anti-correlation—viruses apparently prefer less salty water. The scientist then called over another PI and her student, who had generated profiles counting specific types of virus using a different methodology. We joined the derived data with his data, and then we visualized each individual species against the total virus, without finding a significant relationship. Next, a dataset of the total bacteria counts from a third lab was joined, and this time the graph showed a strong positive correlation, with a few off-diagonal points that had proportionally 'extra' virus. *Without the interactive visualization tools, the analysis would likely have stopped at posing the initial question*, and working out the relationships would probably have been deferred until after the meeting and taken place over email, possibly over days. In contrast, *in a matter of five minutes with our tools*, three PIs and their students had integrated data, discovered two interesting relationships between variables, tested and discarded three other relationships, and identified at least one regime—the 'extra' virus counts—for further investigation.

Interactive quality control Not uncommonly, we noticed that a question could not be answered because data was not collected during the trip or not yet uploaded to SQLShare. In other cases, *we detected a mistake in the query during the visualization process* because certain values did not make sense. This happened once when a scientist imported a multi-joined dataset into ODV. He told us that the data could not be valid, as several stations were missing. Looking at the query, we quickly realized that an inner join had eliminated much of the data, and rewrote the query as an outer join. In another case, swapped latitude and longitude (Section 5) were found when ODV plotted the cruise in the wrong part of the world.

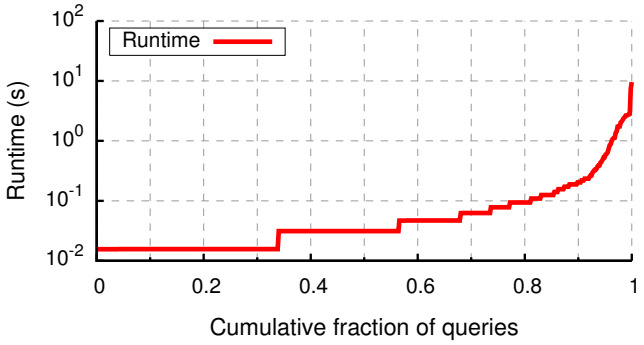
Interactive interdisciplinary cross-validation. Between breakout sessions, each group would assemble its findings and then the leader would present them to the entire meeting. In the morning session on the second day, the within-discipline members of breakout compared and contrasted their findings on "like" data, often integrating it in SQLShare and visualizing the integration to present a unified (or differing) perspective on the phenomena being studied, such as zinc limitation (when not enough zinc is present to satisfy the needs of certain organisms). In the afternoon session, members of different disciplines worked together to integrate their data and refine their findings using data of fundamentally different types, e.g., concentrations of about trace metals with measurements of nutrients. *The ability to quickly and easily integrate disjoin data types together enabled joins of up to 8 different (cleaned) datasets*, e.g., Query 9, and multi-variate analysis to be conducted.

7.4 Results and lessons learned

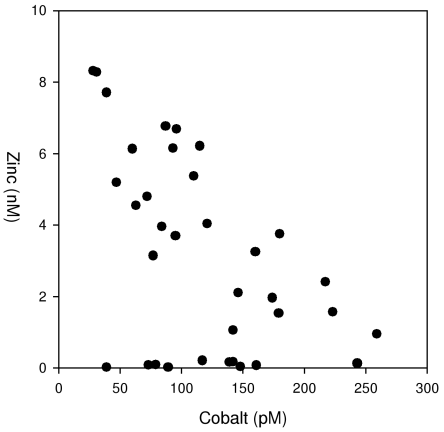
Overall, we generated 39 derived datasets during the workshop, which were used actively during the workshop as well as prepared for later analyses.

What went right:

• One concern was that the SQLShare system processing would be a bottleneck for interactive analysis. Other successful scientific database systems, such as the Sloan Digital Sky Survey [10], use a carefully-engineered schema, indexes, stored procedures, and distributed processing. In contrast, SQLShare runs on a single Microsoft Windows Azure SQL instance, ad hoc schema chosen by the system at runtime, and there are no indexes. Would JOIN-intensive workloads on SQLShare suffer as a result? Figure 8 shows the runtime of 532 queries generated at the data synthesis workshop. A key lesson is that in the *high-variety* regime, queries are fast.



8: The runtime of the queries generated at the data synthesis workshop. This is *high-variety* data, not high-volume data, and in this regime queries are fast. Fully 80% of queries completed within 100 ms, 95% of queries finished in under 1 s, and all queries took less than 10 s to run.



9: This graph, part of a larger multi-variate analysis, shows the relationship between zinc and cobalt (Query 6). That that Zinc and Cobalt are generally anti-correlated is interesting in its own right, and the region at the lower left where Zinc is low indicates 'Zinc stress'. The need to further explore the relationship between Zinc and Cobalt was a key finding of this meeting.

Fully 80% of queries completed within 100 ms, 95% of queries finished in under 1 s, and all queries took less than 10 s to run.

• We found that the ability to integrate data from different datasets and visualize them in real-time was indeed valuable. Almost every graph or result presented at the breakout summary sessions was generated using SQLShare, and many used one of our visualization aids as well. Query 6, the relationship between Zinc and Cobalt concentrations (Figure 9), turned out to be *a key finding of this workshop*. A researcher phrased the finding as follows: "We see an increase of Zinc along an increasing concentration of Cobalt. Except, there's a departure [in the lower left], and it is found in surface water, so the question then becomes, is this a place where we start looking at substitution of Cobalt for Zinc? Diatoms [which live in surface water] prefer Zinc to Cobalt, but when they are Zinc stressed, they turn to Cobalt."

• We found that buy-in from the lead PI was critical—she was able to convince the scientists to use the prepared template keys and to generate questions ahead of time. In this first experiment, these features were important to our success.

• The SQLShare REST interface was invaluable for the visualization tools and for authoring the script-generated queries used to implement advanced features like extracting data from filenames, un-pivoting data in files, and data cleaning. Without these abilities, our SQL "reaction times" would have been much worse.

What went wrong:

• We were not able to answer every question that came up during the workshop. This was not fundamental: principally, it was due to a shortage of time—we could have used at least twice as many data specialists. In other cases it was because the data were not yet available (or so large that the researchers had not completed uploading them when the workshop started).

• There was occasional language mismatch. In the first breakout session, many scientists asked for the correlation between two variables. The lead author, busy answering other queries, deferred this task until a break in order to develop and debug a SQL idiom to compute the ρ^2 coefficient. It turned out the scientist just wanted a plot to visualize the relationship—at least in this community, "correlation" is commonly used to mean visualization.

• The standard deviation of question answering time was high. Once practiced, we could answer many integration queries in less than a minute. Yet, some of the queries took longer because of data cleanup, slow UI response from SQLShare, or work arounds for software bugs. (For instance, many common web clients place a 2000-character limit on URLs. This is problematic when sending script-generated SQL queries over 100s of columns.)

• Neither we nor the scientists were prepared for the problems of integrating -omics data. In particular, the main output of annotated sequence data is human-generated functional annotations. Different types of -omics are aligned against different databases that use different annotation techniques. As a result, asking whether the transcriptomic data contained a gene encoding a particular protein was an un-answerable question—the annotations were unlikely to line up, and there was no common 'key' that disparate data were joined with. At the workshop, one participant described an alternative alignment method based on a clustered database[5] that *does* have a unified keyspace against which different types of -omics can be joined, and hence compared. The -omics participants at the workshop plan to adopt this approach going forward.

Future opportunities:

• Although the REST API allowed two different visualization services to be developed and deployed in a single day, they were important enough to the process to motivate building visualization capabilities natively in SQLShare.

• In this experiment, we did not attempt to have scientists write SQL queries themselves, but the ability to "self-serve" is clearly desirable. A system that allows a complete novice to walk up and write queries of a comparable complexity to the ones we have described with no training represents an ambitious research goal for a collaboration between the

[5] `http://www.ncbi.nlm.nih.gov/proteinclusters`

database community, the HCI community, and the eScience community.

- Our success was assisted by the use of top-down standards to homogenize the data, and some up-front cleaning in the days leading up to the event. In some situations, even these minor luxuries are not available. A system that can achieve similar results with zero assumptions about the data is an important goal.
- The UI performance was more important than we anticipated. Even minor delays can adversely affect the experience for the user and the experts.
- We did not have a control in this experiment; we are not able to conclude that SQL is any more effective than any other approach for this purpose. However, we hypothesize that general purpose languages, workflow tools, and GUI applications would struggle in this context by requiring significant development time or by severely constraining expressiveness, or both. In future work, it would be useful to conduct a controlled experiment to test this hypothesis.

8. RELATED WORK

Google Fusion Tables allows direct upload of data, limited GUI-based queries, a REST API, and rich visualization capabilities. Fusion Tables shares a similar motivation with SQLShare, but cannot express even the routine tasks we encounter working with scientists (multi-key joins, set operators, common user-defined functions) [3].

The Sloan Digital Sky Survey [10] exposed a multi-terabyte astronomy database over the web through public SQL interfaces, and demonstrated that researchers can and will write SQL queries. The effect the SDSS project had on the field is difficult to overstate; a generation of astronomers now learn SQL in their training, and thousands of papers have been written based on accessing data through SQL. Unlike SDSS, we are exploring scenarios where relying on an pre-engineered schema is not feasible, typically because the cost of developing it cannot be amortized over enough time and use, or because the data are too diverse. SQLShare represents an approach to achieve similar results as SDSS for smaller-scale projects that do not have access to significant database programming expertise.

Other public databases such as the Gene Ontology database [2] and NCBI [7] support SQL access either through client tools or specialize web applications. They do not allow users to upload their own datasets.

Galaxy is a popular web-based workflow engine popular in the life sciences that allows users to create and share data processing pipelines. Relational algebra operators are included in the set of algorithms available, implying that users are interested in basic database queries. Galaxy offers no declarative query language and no support for algebraic optimization, making it infeasible for use in the real-time, interactive scenario we describe in this paper. Other workflow systems including VisTrails, Taverna, and Kepler are also not designed for real-time pipeline authoring, and could not automatically accommodate the unusual file formats we encounter in practice. .

OData [8] is a standardized API for accessing and querying data over the web. The OData API supports only basic filtering on individual tables and cannot be used to express joins or other non-trivial queries.

9. CONCLUSIONS

Our hypothesis was that declarative query languages could be used to facilitate interactive, collaborative science even without the benefit of an engineered schema. We tested this hypothesis anecdotally by ingesting heterogeneous science data into a web-based query-as-a-service system called SQLShare and participating in a research meeting with geochemical oceanographers and microbial ecologists, attempting to write queries in real-time in response to the discussion. This experiment was a success: we received highly favorable feedback from the researchers, and authored 39 reusable views representing researchers' hypotheses. The technology and approach fundamentally changed the meeting by allowing scientific Q&A while the collaborators were all present to discuss the findings, as opposed to working independently when they returned to their labs.

10. REFERENCES

[1] EAGER: The relationship between microbial biogeography and ocean chemistry across a persistent oceanographic "hot spot" in the NE Pacific Ocean. `http://www.nsf.gov/awardsearch/showAward?AWD_ID=1205233`.

[2] The Gene Ontology Database. `http://www.geneontology.org/`.

[3] H. Gonzalez, A. Y. Halevy, C. S. Jensen, A. Langen, J. Madhavan, R. Shapley, W. Shen, and J. Goldberg-Kidon. Google Fusion Tables: web-centered data management and collaboration. In *SIGMOD Conference*, pages 1061–1066, 2010.

[4] J. Gray and A. S. Szalay. Where the rubber meets the sky: Bridging the gap between databases and science. *CoRR*, abs/cs/0502011, 2005.

[5] B. Howe, G. Cole, E. Souroush, P. Koutris, A. Key, N. Khoussainova, and L. Battle. Database-as-a-service for long tail science. In *SSDBM '11: Proceedings of the 23rd Scientific and Statistical Database Management Conference*, 2011.

[6] B. Howe, F. Ribalet, D. Halperin, S. Chitnis, and E. V. Armbrust. SQLShare: Scientific workflow via relational view sharing. *Computing in Science & Engineering, Special Issue on Science Data Management*, 15(2), May/June 2013.

[7] National Center for Biotechnology Information. `http://www.ncbi.nlm.nih.gov/`.

[8] Open Data Protocol. `http://www.odata.org/`.

[9] J. Rogers, R. Simakov, E. Soroush, P. Velikhov, M. Balazinska, D. DeWitt, B. Heath, D. Maier, S. Madden, J. Patel, M. Stonebraker, S. Zdonik, A. Smirnov, K. Knizhnik, and P. G. Brown. Overview of SciDB: Large scale array storage, processing and analysis. In *Proc. of the SIGMOD Conf.*, 2010.

[10] Sloan Digital Sky Survey. `http://cas.sdss.org`.

[11] J. E. Swalwell, F. Ribalet, and E. V. Armbrust. SeaFlow: A novel underway flow-cytometer for continuous observations of phytoplankton in the ocean. *Limnology & Oceanography Methods*, 9:466–477, 2011.