Even Metadata is Getting Big: Annotation Summarization using InsightNotes *

Dongqing Xiao, Armir Bashllari, Tyler Menard, and Mohamed Eltabakh
Computer Science Department, Worcester Polytechnic Institute
Worcester, MA, US
dxiao@cs.wpi.edu, abashllari@cs.wpi.edu, tjmenard@cs.wpi.edu, meltabakh@cs.wpi.edu

ABSTRACT

In this paper, we demonstrate the *InsightNotes* system, a *summary-based annotation management engine over relational databases* [30]. InsightNotes addresses the unique challenges that arise in modern applications—especially scientific applications—that rely on rich and large-scale repositories of curation and annotation information. In these applications, the number and size of the raw annotations may grow beyond what end-users and scientists can comprehend and analyze. InsightNotes overcomes these limitations by integrating mining and summarization techniques with the annotation management engine in novel ways. The objective is to create concise and meaningful representations of the raw annotations, called *"annotation summaries"*, to be the basic unit of processing. The core functionalities of InsightNotes include: **(1) Extensibility**, where domain experts can define the summary types suitable for their application, **(2) Incremental Maintenance**, where the system efficiently maintains the annotation summaries under the continuous addition of new annotations, **(3) Summary-Aware Query Processing and Propagation**, where the execution engine and query operators are extended for manipulating and propagating the annotation summaries within the query pipeline under complex transformations, and **(4) Zoom-in Query Processing**, where end-users can interactively expand specific annotation summaries of interest and retrieve their detailed (raw) annotations. We will demonstrate the InsightNotes's features using a real-world annotated database from the ornithological domain (the science of studying birds). We will design an interactive demonstration that engage the audience in annotating the data, visualizing how annotations are summarized and propagated, and zooming-in when desired to retrieve more details.

1. INTRODUCTION

The virtue and merit of data curation and annotation is becoming increasingly important in modern applications. This is evident from scientific applications that collect and generate metadata information in the scale of orders-of-magnitudes larger than the base data, e.g., the number of annotations is around 30x, 120x, and 250x larger than the number of data records in DataBank biological database [2], Hydrologic Earth database [3, 29], and AKN ornithological database [4], respectively. Annotations may range from capturing scientists' observations about the data, attaching related articles or documents, exchanging auxiliary knowledge among users, to highlighting erroneous or conflicting values, and storing provenance and lineage information. Existing techniques in annotation management, e.g., [6, 10, 11, 15, 17, 20], have made it feasible to systematically capture such metadata annotations and efficiently integrate them into the data processing cycle. This includes propagating the related annotations along with queries' answers [6, 10, 11, 20, 28], querying the data based on their attached annotations [15, 20], annotation-based scientific workflows [5, 7, 25], and supporting semantic annotations such as provenance tracking [8, 9, 14, 27], and belief annotations [19].

Why Annotations are Not Data: Annotations are logically different from the base data since they are viewed as metadata information that should propagate automatically with the data. Since the data values may go though complex transformations during query processing, e.g., projection, join, grouping and aggregation, and duplicate elimination, the related annotations must also go though corresponding transformations by each query operator. If annotations are modeled as regular data, then the annotation management tasks are entirely delegated to end-users and higher-level applications starting from the storage and indexing of annotations and ending by explicitly encoding the propagation semantics within each of the users' queries (An example illustrating the complexity of annotation propagation is presented in Section 2.1). Evidently, this approach is not only error-prune, lacks optimizations, not feasible in some applications, but also render even simple queries very complex. That is why annotation management engines have been proposed to efficiently and transparently manage such complexities.

In this paper, we propose to demonstrate the *"InsightNotes"* system, *an advanced summary-based annotation management engine over relational databases* [22, 30]. InsightNotes overcomes a critical and common limitation to all existing techniques in annotation management, which is that they all manage and manipulate the raw annotations. As a result—under the current large repositories of annotations—a single output tuple may have 100s of raw annotations attached to it. For example, the L.H.S of Figure 1 illustrates a single data tuple having 100s of raw annotations attached to it. Hence, it is extremely hard for scientists to analyze the reported annotations and extract useful knowledge from them, e.g., finding out which ones carry provenance information vs. regular comments, which ones are obsolete or proven wrong, which ones are duplicates or have similar content, and which ones refute (or approve) the content of their tuples.

*This project is partially supported by NSF-CRI 1305258 grant.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGMOD'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.
Copyright © 2015 ACM 978-1-4503-2758-9/15/05 ...$15.00.
http://dx.doi.org/10.1145/2723372.2735355.

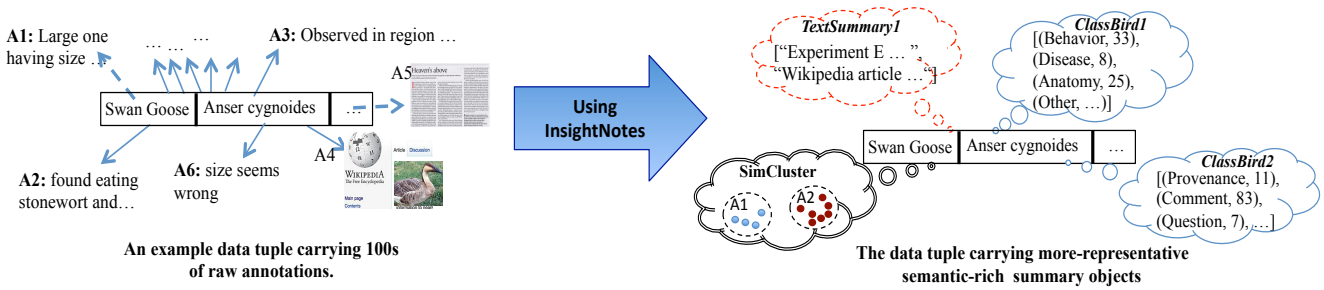

Figure 1: Example of Annotation Summaries in InsightNotes.

InsightNotes is fundamentally different from the existing techniques as it proposes a new annotation management model based on the new concept of *"annotation summaries"*. InsightNotes integrates data mining and summarization techniques with the annotation management engine. The objective is to create concise and meaningful representations of the raw annotations, i.e., the *annotation summaries*, to be the basic unit of processing and propagation. For example, the R.H.S of Figure 1 illustrates the same data tuple along with its attached summary objects using InsightNotes. The summary objects include, for example, Classifier-type objects, e.g., *ClassBird1* and *ClassBird2*, that classify the raw annotations into user-defined classes, Snippet-type objects, e.g., *TextSummary1*, that summarize the attached big articles and report snippets on each, and Cluster-type objects, e.g., *SimCluster*, that group similar annotations into groups and reports only one representative from each group. It is worth highlighting that these summary objects are created per data tuple, i.e., they summarize the annotations on a single tuple, and then they get attached back to this tuple as depicted in the figure.

In the demonstration, we will present the novel features and capabilities of the *InsightNotes* engine that enable the creation, manipulation, and propagation of the annotation summaries. These features include:

(1) Extensibility and Efficient Maintenance: InsightNotes is designed as an extensible engine where the database admins can define how to summarize the annotations in a way suitable for their applications, e.g., determining which classification techniques to use and what class labels to generate. The system only defines some properties and requirements that the integrated mining techniques should obey for efficient and incremental maintenance of the annotation summaries.

(2) Summary-Aware Query Processing: Unlike the raw annotations which are free-text objects, the annotation summaries have well-defined structures and properties. The challenge is how to extend the query engine to efficiently manipulate the annotation summaries at query time without retrieving the raw annotations. The semantics and algebra of each of the standard query operators, e.g., projection, join, grouping, and aggregation, have been extended to directly operate on the summary objects attached to each tuple. We also introduced new query operators specific for annotation summaries and integrated them into the query engine.

(3) Zoom-in Query Processing: After reporting the annotation summaries, end-users may get interested in retrieving the detailed (raw) annotations of specific summaries. For example, referring to the R.H.S of Figure 1, a scientist may be interested in retrieving the eight disease-related annotations attached to the given tuple, or in retrieving all annotations in the cluster represented by annotation *A2*. This feature is enabled in InsightNotes through an interactive *zoom-in* querying capability. We will demonstrate this feature coupled with smart caching techniques for efficient execution.

(4) InsightNotes Interface: Our team is developing an Excel-based GUI on top of the InsightNotes engine from which most of the proposed functionalities can be realized. We opt for an Excel-based tool since scientists are already familiar with Excel and its functionalities. The tool will enable, querying the data interactively, visualizing the reported annotation summaries, and optionally zooming-in to retrieve the detailed raw annotations.

2. InsightNotes OVERVIEW

In this section, we briefly highlight the core features of the InsightNotes system. The system has an extended data model, where each data tuple r carries its attribute values as well as the annotation summary objects that summarize the raw annotations on r (See Figure 1). InsightNotes supports three widely-used families (types) of data mining and summarization techniques, which are: (1) Text summarization techniques (Snippets), e.g., [24], for summarizing large-object annotations, e.g., big text values and large documents, and creating concise snippets from them, (2) Clustering techniques, e.g., [23], for clustering the annotations into distinct groups of similar content, and (3) Classification techniques, e.g., [12], for categorizing annotations according to user-defined classifiers. In the following, we highlight the query processing, extensibility, and scalability features of InsightNotes.

2.1 Summary-Aware Query Processing and Propagation

InsightNotes has an extended query engine for manipulating the annotation summaries attached to each data tuple under complex transformations, e.g., join, grouping, projection, and duplicate elimination. We proposed extended semantics for each query operator to compute the output annotation summaries from the input values without accessing the raw annotations. A key contribution of InsightNotes is that the summaries are manipulated in a pipelined fashion. As a result, summary-based processing can be plugged-in at any stage of the query plan, e.g., filtering, joining, or sorting the data tuples according to summary-based predicates. The following example demonstrates a Select-Project-Join (SPJ) query involving summary propagation in InsightNotes. The formal semantics of all query operators can be found in [30].

Example: *Assume an SQL query* `"Select r.a, r.b, s.z From R r, S s Where r.a = s.x And r.b = 2"` *over the two tuples r and s presented in Figure 2. Tuple r has four summary objects attached to it, while tuple s has only two attached summary objects. We proved in [30] in Theorems 1 and 2 that to guarantee identical summary propagation under different—but equivalent—query plans, InsightNotes needs to project out the un-needed annotations before any merge operation over the summary objects. Therefore, the projection operator in Step 1 in Figure 2 projects out attributes $r.c$ and $r.d$ and eliminates*

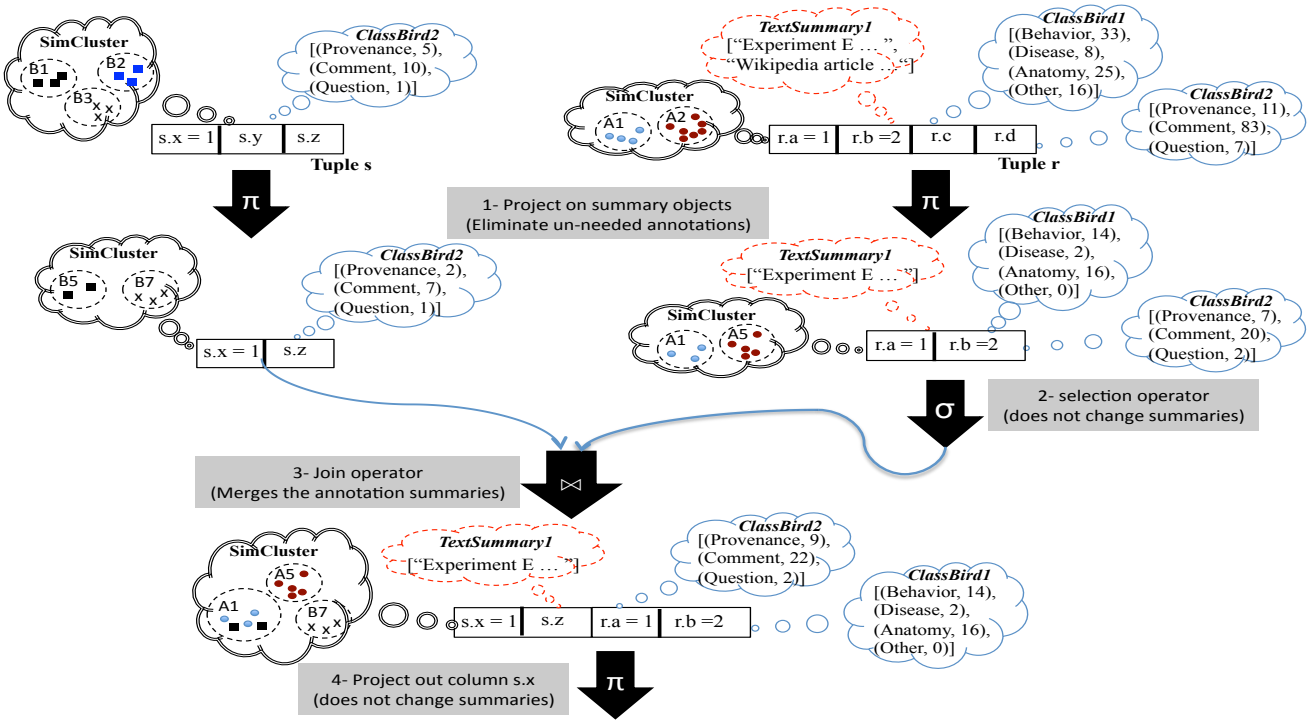

Figure 2: Example Query in InsightNotes.

the effect of their annotations from r's summary objects. For example, the `annotationCnt` *field in the classifier objects is decremented, the wikipedia article in the snippet object is deleted, and the cluster objects are modified, e.g., some annotations are dropped from each cluster, and hence the* `groupSize` *field is decremented. Moreover, if a cluster's representative is dropped, then another representative is elected (See $A5$ representative replacing the dropped $A2$ representative). The same operation takes place over tuple s, where the effect of all annotations attached to both $s.x$ and $s.y$ is removed from s's summary objects. The only difference is that $s.x$ attribute will not be projected out because it is needed in the subsequent join operator.*

The next operator in the query plan is the selection operator over r (Step 2). Based on the query's predicate, r will pass the operator and all its summary objects will propagate without any change. Then, the produced tuples will join and their summary objects will be merged (Step 3). According to the merge procedure, r's summary objects `ClassBird1` *and* `TextSummary1` *will propagate without any change since they do have no counterpart objects over s. Whereas summary objects* `ClassBird2` *and* `SimCluster` *will be combined. This action takes into account the case where the same annotation may be attached to both tuples r and s, and hence the annotation's effect on the summary objects should not be double counted. For example, assuming that there are five common annotations on both r and s classified as* `Comment`*, then when the two objects are merged the sum of that classifier label will be 22 instead of 27 as illustrated in the figure. The merge of the cluster summary objects is slightly more complex. The main idea is that the overlapping groups from both sides, e.g., the groups represented by $A1$ and $B5$, will be combined together, whereas the non-overlapping groups, e.g., the groups represented by $A5$ and $B7$, will propagate separately as illustrated in the figure. Finally attribute $s.x$ will be projected out before producing the output.*

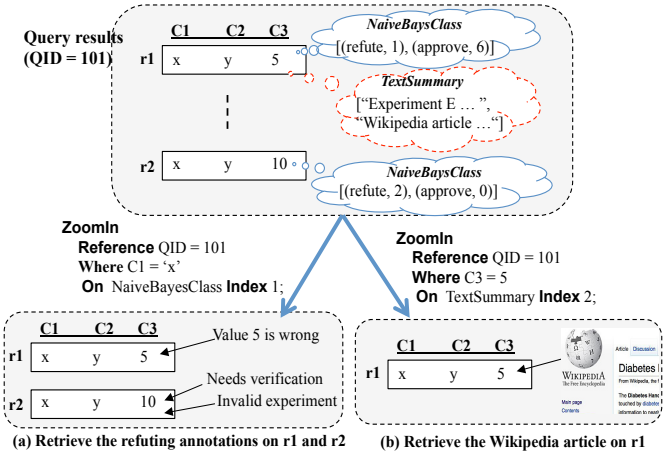

Figure 3: Zoom-In Query Processing.

2.2 Zoom-In Query Processing

After receiving the results of a query along with the attached annotation summaries, end-users may investigate the propagated summaries and get interested in zooming-in and retrieving more details about specific summaries (See Figure 3). The zoom-in capability in InsightNotes enables performing this operation interactively and efficiently. Users can reference queries that they have just executed (using unique QIDs assigned to the result), refine which tuples from the result they want to focus on (using predicates in the `ZoomIn` command), and then specify which summaries to retrieve their details. For example, in Figure 3(a), the `ZoomIn` command selects tuples r_1 and r_2, and retrieves the actual annotations refuting (disapproving) the content of these tuples (one anno-

tation on r_1, and two annotations on r_2). The `ZoomIn` command specifies in the `ON` clause that the zoom-in operation is over the classifier summary object of type *NaiveBayesClass*, and the index of "1" indicates the 1^{st} class label within the object, which is the *"refute"* label. Similarly, the second command, retrieves the complete Wikipedia article attached to r_1.

In addition to demonstrating the high-level feature, we will also demonstrate the underlying caching and materialization techniques behind the efficient execution of the zoom-in operation. For example, we proposed a materialization technique, which allows the queries' results to compete with each other over a limited disk-based cache—where they are temporarily kept to serve future zoom-in operations. Adding to (or evicting from) the cache is controlled by a new replacement policy, called *RCO* (stands for Recency, Complexity, and Overhead), that takes into account three factors for replacement, i.e., query complexity and cost, the results' size, and how frequently and recently the results have been referenced in a zoom-in operation. The demonstration will illustrate the effect of the cache on the zoom-in performance.

2.3 Extensibility & Scalable Maintenance

Different applications may maintain and manage annotations of different semantics, and hence they may target different types of summarization outputs. For example, in biological databases, it can be meaningful to classify the annotations attached to the gene tuples into the classes of {'FunctionPrediction', 'Provenance', 'Comment'}, while in ornithological databases it can be more meaningful to classify them into the classes of {'Behavior', 'Disease', 'Anatomy', 'Other'}. To achieve broader applicability, InsightNotes is designed as an extensible engine, where it only defines some properties and requirements that the integrated summarization techniques should obey, while the rest is customizable by domain experts and database admins.

Figure 4 illustrates the three-level summarization hierarchy of InsightNotes. The 1^{st} level is the *Summary Types*, where the summarization types of *Cluster*, *Classifier*, and *Snippet* are integrated within the query engine and any instance of these types can be supported. The 2^{nd} level is the *Summary Instances*, where domain experts and database admins can fully customize the desired instances, e.g., for a *Classifier* type, the instance defines the underlying algorithm to use, the output class labels, configuration parameters, and training datasets or classifier model. In this level, domain knowledge, e.g., ontologies for the available annotations, can be also integrated with the summarization technique. Finally, the created instances can be linked in a many-to-many relationship with users' relations. If an instance is linked to a given relation, say R, then the annotations on each of R's tuples will be summarizes by that instance. The summarization output creates the 3^{rd} level of the hierarchy, which is the *Summary Objects*, and each tuple will carry these summary objects along with its data during the query execution.

Despite the extensibility and flexibility of InsightNotes, the system deploys various optimizations for efficient incremental maintenance of the annotation summaries. These optimizations enable the system to achieve scalability w.r.t both the number of raw annotations attached to the data and the number of summary instances defined on top of them. One example of these optimizations is controlled by the *Properties* field within the summary instance (See Figure 4). The Boolean *AnnotationInvariant* property indicates whether or not the summarization of a newly inserted annotation $\mathbb{a}$ over tuple t depends on t's current annotations. In contrast, the Boolean *DataInvariant* property indicates whether or not the summarization depends on t's content, i.e., the data values. If both

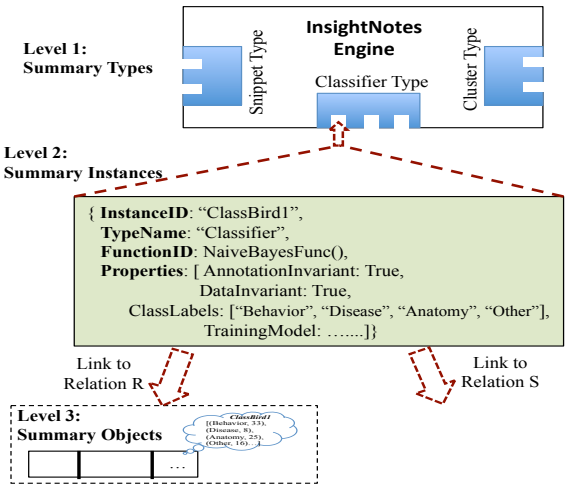

Figure 4: Extensibility and Summarization Hierarchy.

properties are True, then the summarization depends on neither of t's annotations nor t's content. Therefore, the system will summarize $\mathbb{a}$ only once even if it will attached to many data tuples.

3. DEMONSTRATION SCENARIOS

Interactive Demo using Ornithological Datasets: We plan to develop an interactive audience-engaged demonstration using real-world datasets obtained from the AKN system. In the AKN system, in addition to the base data, i.e., the birds' basic information such as scientific names, synonyms, geographic ranges, images, description, etc., there are more than 200,000 bird watchers and scientists who continuously provide observations and annotations on these birds. It is reported in [1] that, on average, bird watchers add 1.6 million annotations per month to the ebirds system, which is part of the AKN network. These annotations are free-text values that may describe anything related to the observed birds, e.g., color, body shape or weight, certain behavior or sound, eating habits, geographic location, or observed diseases. Therefore, in these databases the number of annotations can be more than two orders of magnitude larger than the number of data records. To encourage the audience's participation and engagement in annotating the data, we will select few widely-known birds to be the target dataset. The dataset set will be already annotated with various types of annotations and related documents, and then the audience can add to that.

InsightNotesGate & Demonstration Features: Our team is developing an Excel-based frontend tool, called *InsightNotesGate*, for visualizing the data and the queries' results as well as the annotation summaries (See Figure 5). We choose an Excel-based GUI because most scientists are already familiar with Excel-based tools and their functionalities. All of the new features and functionalities of the InsightNotes system will be performed through this GUI. The demonstration will involve:

(1) *Querying and Visualizing Summaries:* As illustrated in Figure 5, we developed a customized ribbon in InsightNotesGate that enables end-users to query their database either by entering an explicit SQL statement, or by filling-in fields in a Query-By-Example (QBE) section. The QBE mechanism is more user-friendly, but limited only to select-project queries. In contrast, the explicit SQL mechanism is more flexible as it allows for expressing more complex queries, e.g., join and aggregation. The audience can spec-

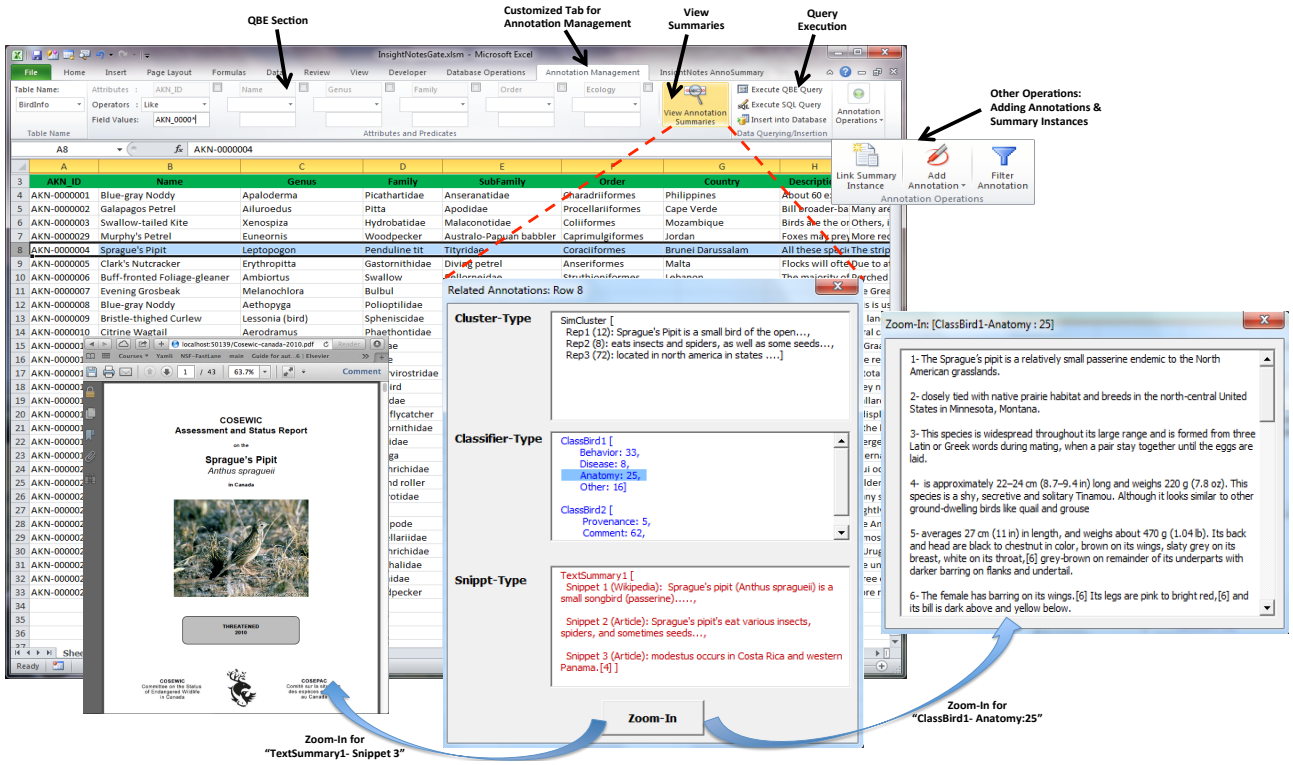

Figure 5: InsightNotesGate: InsightNotes Interactive Excel-Based GUI.

ify their query predicates in the QBE section, and then submit the query for execution. The resultset will be displayed in the excel sheet as depicted in the figure. By highlighting a specific data row, the attached annotation summaries can be visualized by clicking the button *"Visualize Annotation Summaries"*. The annotation summaries will be displayed in a separate window, and categorized into three sections, *Cluster-Type*, *Snippet-Type*, and *Classifier-Type*, as depicted in Figure 5. Recall that multiple summary instances of each type can be defined and linked to a single user's relation. For example, in the visualized scenario, there are two Classifier-Type, one Cluster-Type, and one Snippet-Type instances linked to the database table.

Audience will have the ability to examine the reported summaries, select specific objects, and then apply a zoom-in operation to retrieve the detailed raw annotations. For example, referring to Figure 5, by selecting the class label ("Anatomy : 25") within the *ClassBird1* summary object and clicking the *Zoom-In* button, the system will retrieve the detailed 25 raw annotations assigned to this label for display. Similarly, by selecting the 3^{rd} snippet within the *TextSummary1* object and clicking the *Zoom-In* button, the system will retrieve the corresponding article for display.

(2) *Adding Annotations and Summary Instances:* The audience will be encouraged to add more annotations over the data. This feature is supported from InsightNotesGate by selecting specific table cell(s) or entire data row(s) from the displayed dataset, and then clicking the *"Add Annotation"* button (See Figure 5). A separate window will be opened allowing the insertion of free-text comment, or attaching auxiliary documents or files. The new annotation will be reflected to the database, update the existing summaries, and then the visualized resultset will be refreshed. To demonstrate the system's extensibility, the tool will also enable creating new links (or dropping existing links) between summary instances and users' relations, e.g., linking a new Classifier-type or Snippet-Type instances to the database table. As a result, the maintained and visualized summary objects will differ accordingly.

(3) *Under-the-Hood Execution:* In addition to the high-level features, an interesting experience for audience is to get exposed to the manipulation of annotation summaries within a query pipeline (e.g., similar to the illustration in Figure 2). Our plan is to design a query that involves some of the basic SQL operators, e.g., selection, join, projection, and duplicate elimination, and visualize its query tree. We then extend the InsightNotes's engine to allow the query operators to log and report their intermediate data, i.e., the data tuples along with their attached annotation summaries. This log information can be then visualized on the query tree.

4. RELATED WORK

The advances in data management and the increasing ability of applications to capture more metadata information have triggered the need for more feature-rich annotation management engines. On one hand, end-users and scientists are incapable of analyzing and extracting knowledge from the large number of reported annotations. On the other hand, the current annotation management techniques fall short in providing advanced processing over the annotations beyond just propagating them to end-users [6, 11, 15, 20, 21, 28]. The InsightNotes system has been proposed to bridge this gap by providing real insights on top of the raw annotations. Therefore, InsightNotes addresses several unique challenges that have not been addressed by existing techniques.

Scientific systems and workflow management techniques have also leveraged the concept of semantic and ontology-based annotations, e.g., [5, 7, 16, 25]. For example, the work in [25] uses the annotations drawn from a specific ontology to relate and measure

the similarity among the scientific entities in the database. And hence, it uses the annotations as a similarity metric among the data entires. In contrast, the work in [5, 7] uses semantic annotations to either summarize complex workflows [5], or help in building and verifying workflows [7]. These systems are based on workflow- and process-centric annotations, e.g., annotations capturing the semantics of each function in a workflow, the structure of their input and output arguments, etc. In contrast, InsightNotes manages data-centric annotations that are independent from how the data is processed.

In the domains of e-commerce, social networks, and entertainment systems, e.g., [18], the annotations are usually referred to as *tags*. These systems deploy advanced mining and summarization techniques for extracting the best insight possible from the annotations to enhance users' experience. They use such extracted knowledge to take actions, e.g., providing recommendations and targeted advertisements [13, 26]. However, unlike relational DBs, the retrieval mechanisms in these systems are typically straightforward and do not involve complex processing or transformations, i.e., objects (products in Amazon or movies in Netflix) are usually queried as individual instances without going through a complex pipeline of query operators, e.g., projection, join, grouping, and aggregation operators. Therefore, no advanced query processing is required over the annotations summaries once created.

5. REFERENCES

[1] eBird Trail Tracker Puts Millions of Eyes on the Sky. `https://www.fws.gov/refuges/RefugeUpdate/MayJune_2011/ebirdtrailtracker.html`.

[2] Gene Ontology Consortium. `http://geneontology.org`.

[3] Hydrologic Information System CUAHSI-HIS. `(http://his.cuahsi.org`.

[4] The Avian Knowledge Network (AKN). `http://www.avianknowledge.net/`.

[5] P. Alper, K. Belhajjame, C. Goble, and P. Karagoz. Small Is Beautiful: Summarizing Scientific Workflows Using Semantic Annotations. In *IEEE BigData Congress*, pages 318–325, 2013.

[6] D. Bhagwat, L. Chiticariu, and W. Tan. An annotation management system for relational databases. In *VLDB*, pages 900–911, 2004.

[7] S. Bowers and B. LudŁscher. A Calculus for Propagating Semantic Annotations through Scientific Workflow Queries. In *In Query Languages and Query Processing (QLQP)*, 2006.

[8] P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *SIGMOD*, pages 539–550, 2006.

[9] P. Buneman, S. Khanna, and W. Tan. Why and where: A characterization of data provenance. *Lec. Notes in Comp. Sci.*, 1973:316–333, 2001.

[10] P. Buneman, E. V. Kostylev, and S. Vansummeren. Annotations are relative. In *Proceedings of the 16th International Conference on Database Theory*, ICDT '13, pages 177–188, 2013.

[11] L. Chiticariu, W.-C. Tan, and G. Vijayvargiya. DBNotes: a post-it system for relational databases based on provenance. In *SIGMOD*, pages 942–944, 2005.

[12] P. R. Christopher D. Manning and H. Schutze. Book Chapter: Text classification and Naive Bayes, in Introduction to Information Retrieval. In *Cambridge University Press*, pages 253–287, 2008.

[13] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web (WWW)*, pages 271–280, 2007.

[14] S. B. Davidson and J. Freire. Provenance and scientific workflows: challenges and opportunities. In *SIGMOD*, pages 1345–1350, 2008.

[15] M. Eltabakh, W. Aref, A. Elmagarmid, and M. Ouzzani. Supporting annotations on relations. In *EDBT*, pages 379–390, 2009.

[16] M. Eltabakh, M. Ouzzani, and W. Aref. bdbms-database management system for biological data. In *CIDR*, pages 196–206, 2007.

[17] M. Y. Eltabakh, W. G. Aref, and A. K. Elmagarmid. A database server for next-generation scientific data management. In *ICDE Workshops*, pages 313–316, 2010.

[18] A. Gattani and et. al. Entity extraction, linking, classification, and tagging for social media: a wikipedia-based approach. *Proc. VLDB Endow.*, 6(11):1126–1137, 2013.

[19] W. Gatterbauer, M. Balazinska, N. Khoussainova, and D. Suciu. Believe it or not: adding belief annotations to databases. *Proc. VLDB Endow.*, 2(1):1–12, 2009.

[20] F. Geerts and et. al. Mondrian: Annotating and querying databases through colors and blocks. In *ICDE*, pages 82–93, 2006.

[21] F. Geerts and J. Van Den Bussche. Relational completeness of query languages for annotated databases. In *Proceedings of the 11th international conference on Database Programming Languages (DBPL)*, pages 127–137, 2007.

[22] K. Ibrahim, D. Xiao, and M. Y. Eltabakh. Elevating Annotation Summaries To First-Class Citizens In InsightNotes. In *EDBT Conference*, 2015.

[23] Y.-B. Liu, J.-R. Cai, J. Yin, and A. W. Fu. Clustering Text Data Streams. *Journal of Computer Science and Technology*, 23(1):112–128, 2008.

[24] A. Nenkova and K. McKeown. A Survey of Text Summarization Techniques. In *Book: Mining Text Data*, pages 43–76, 2012.

[25] G. Palma and et. al. Measuring Relatedness Between Scientific Entities in Annotation Datasets. In *International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, pages 367:367–367:376, 2013.

[26] A. Rae, B. Sigurbjörnsson, and R. van Zwol. Improving tag recommendation using social networks. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, RIAO, pages 92–99, 2010.

[27] Y. L. Simmhan, B. Plale, and D. Gannon. A survey of data provenance in e-science. *SIGMOD Record*, 34(3):31–36, 2005.

[28] W.-C. Tan. Containment of relational queries with annotation propagation. In *DBPL*, 2003.

[29] D. Tarboton, J. Horsburgh, and D. Maidment. CUAHSI Community Observations Data Model (ODM),Version 1.1, Design Specifications. In *Design Document*, 2008.

[30] D. Xiao and M. Y. Eltabakh. InsightNotes: Summary-Based Annotation Management in Relational Databases. In *SIGMOD Conference*, pages 661–672, 2014.