Ersch. in: DOLAP'12 Proceedings of the fifteenth international workshop on Data warehousing and OLAP / Il-Yeol Song, Matteo Golfarelli (eds.). - New York, NY : ACM, 2012. - S. 9-16. - ISBN 978-1-4503-1721-4

http://dx.doi.org/10.1145/2390045.2390048

Discovering OLAP Dimensions in Semi-Structured Data

Svetlana Mansmann
University of Konstanz
Konstanz, Germany
svetlana.mansmann@uni-konstanz.de

Nafees Ur Rehman
University of Konstanz
Konstanz, Germany
nafees.rehman@uni-konstanz.de

Andreas Weiler
University of Konstanz
Konstanz, Germany
andreas.weiler@uni-konstanz.de

Marc H. Scholl
University of Konstanz
Konstanz, Germany
marc.scholl@uni-konstanz.de

ABSTRACT

OLAP cubes are obtained from the input data based on the available attributes and known relationships between them. Transforming the input into a set of measures distributed along a set of uniformly structured dimensions may be unrealistic for applications dealing with semi-structured data. We propose to extend the capabilities of OLAP via content-driven discovery of measures and dimensions in the original dataset. New elements are discovered by means of data mining and other techniques and are therefore expected to be of limited temporal validity. In this work we focus on the challenge of generating, maintaining, and querying such discovered elements of the cube.

We demonstrate the power of our approach by providing OLAP to the public stream of user-generated content provided by Twitter. We were able to enrich the original set with dynamic characteristics such as user activity, popularity, messaging behavior, as well as to classify messages by topic, impact, origin, method of generation, etc. Knowledge discovery techniques coupled with human expertise enable structural enrichment of the original data beyond the scope of the existing methods for obtaining multidimensional models from relational or semi-structured data.

Categories and Subject Descriptors

H.2.7 [**Database Management**]: Database AdministrationData warehouse and repository; H.2.1 [**Database Management**]: Logical Design—*data models*

Keywords

OLAP, multidimensional data model, semi-structured data

1. INTRODUCTION AND MOTIVATION

Explosion of social network activity in the recent years has lead to generation of massive volumes of user-related data, such as status updates, messaging, blog and forum entries and has given birth to novel areas of data analysis, such as *Social Media Analysis* and *Social Network Analysis*. This phenomenon can be viewed as a part of the *"Big Data"* [25] challenge, which is to cope with the rising flood of digital data from many sources, including mobile phones, internet, videos, e-mails, and social network communication. The generated content is heterogeneous and encompasses textual, numeric, and multimedia data. Companies and institutions worldwide anticipate to gain valuable insights from big data and hope to improve their marketing, customer services and public relations with the help of the acquired knowledge. Meanwhile, results of big data analysis are incorporated into e-commerce sites and social networks themselves in the form of personalized content, such as recommendations, suggestions, advertisement, etc.

The established data warehousing technology with On-Line Analytical Processing (OLAP) and data mining (DM)) functionality is known for its universality and high performance, but also for its rigidness and limitations when it comes to semi-structured or complex data. Various solutions have been proposed in theory and practice for warehousing and analyzing heterogeneous data volumes. One class of solutions focuses on extending the capabilities of the predominant technologies, i.e., relational and multidimensional databases, while others pursue novel paths. A prominent example of the latter class is the NoSQL movement that announces the end of the relational era and proposes a wide range of alternative database approaches [27]. Our work fits into the former class since our goal is to adapt the mature OLAP technology to non-conforming data scenarios. Our approach is based on identifying parts of the dataset that can be transformed to facts and dimensions, enriching the outcome by including external services (e.g., language and location recognition tools) and, finally, extending the obtained structures via content-driven discovery of additional data characteristics. The benefit of obtaining a properly structured and consolidated data set is that the latter can be explored with existing tools for data analysis, visualization and mining and be used for a variety of analysis tasks.

The remainder of the introduction is dedicated to the main components of our solution, namely OLAP, data warehousing and mining as the employed data analysis technology and the Twitter social network and its APIs as the underlying data source for building a data warehouse.

1.1 Coupling OLAP and DM

The necessity to integrate OLAP and DM was postulated in the late 90-es [8]. Meanwhile, a powerful data mining toolkit is offered as an integrated component of any mature data warehouse system, such as Microsoft SQL Server, IBM DB2 Data Warehouse Edition, Oracle, and others. DM tools require the input data to be consolidated, consistent and clean. OLAP cubes – where the extracted data undergoes exactly this kind of transformation – appear to be perfect candidates for feeding the DM algorithms. Mining data cubes for dynamic classifications is a popular technique in OLAP applications dealing with customer trending, risk or popularity assessment, etc. However, traditional DM applications return such classifications as the outcome of the analysis, whereas our approach is to feed this outcome back to the data warehouse as elements of the data model in their own right. Introduction of discovered classifications to dimensional hierarchies raises a number of research challenges, such as their maintenance, evolution, temporal validity and aggregation constraints. These issues will be handled later on in this work.

1.2 Tweet Analysis as Motivating Example

Twitter is an outstanding phenomenon in the landscape of social networking. Initially introduced in 2006 as a simple platform for exchanging short messages on the Internet, Twitter rapidly gained worldwide popularity and has evolved into an extremely influential channel of broadcasting news and the means of real-time information exchange. It has revolutionized the culture of interacting and exchanging information on the Internet and impacted various areas of human activity, such as organization and execution of political actions, crime prevention, disaster management, emergency services, etc. Apart from its attractiveness as a means of communication – with over 140 million active users generating over 340 millions tweets daily as of 2012 [29] – Twitter has also succeeded in drawing the attention of political, commercial, research and other establishments by making its data stream available to the public. Twitter provides the developer community with a set of APIs[1] for retrieving the data about its users and their communication, including the *Streaming API* for data-intensive applications, the *Search API* for querying and filtering the messaging content, and the *REST API* for accessing the core primitives of the Twitter platform.

To understand what type of knowledge can be discovered from this data, it is important to investigate the underlying data model. In a nutshell, it encompasses users, their messages (*tweets*), and the relationships between and within those two classes. Users can be friends or followers of other users, be referenced (i.e., tagged) in tweets, be authors of tweets or retweet other users' messages. The third component is the timeline, which describes the evolution, or the ordering, of user and tweet objects. Using the terminology of the Twitter Developer Documentation [15], the data model consists of the following three object classes:

1. **Status Objects** (tweets) consist of the text, the author and their metadata.

2. **User Objects** capture various user characteristics (nickname, avatar, etc.).

3. **Timelines** provide an accumulated view on the user's activity, such as the tweets authored by or mentioning (tagging) a particular user, status updates, follower and friendship relationships, re-tweets, etc.

[1]`https://dev.twitter.com/start`

Even though the above model is not tailored towards OLAP, the offered data perspective is rather suitable for multidimensional aggregation. Essentially, Twitter accumulates various user and message related data over time. With a reasonable effort, this data stream can be transformed into a set of OLAP cubes with a fully automated ETL routine. What makes Twitter a particularly interesting motivating example for introducing the DM feedback loop is the fact that the structure of the original stream explicitly contains a rather small number of attributes usable as measures and dimensions of a cube, whereas a wealth of additional parameters, categories and hierarchies can be obtained using data enrichment methods of arbitrary complexity, from simple computations to complex techniques of knowledge discovery. Many of the characteristics (e.g., status, activity, interests, popularity, etc.) are dynamic and, therefore, cannot be captured as OLAP dimensions by definition. However, from the analyst's perspective, such characteristics may represent valuable dimensions of analysis.

The dataset delivered by the Twitter Streaming API is semi-structured using the JSON (JavaScript Object Notation) as its output format. Each tweet is streamed as a JSON object containing 67 data fields with high degree of heterogeneity. A tweet record encompasses the tweeted message itself along with detailed metadata on the user's profile and geographic location. A straightforward mapping of this set of attributes to a multidimensional perspective results in the identification of cubes *Tweet* and *TweetCounters* for storing the contents and the metadata of the messages and for storing the statistical measurements provided with each record, respectively.

1.3 Related Work

The work related to our contribution can be subdivided into three major sections: 1) integrating data warehousing and mining, 2) OLAP for complex data, and 3) social network data analysis.

A pioneering work on integrating OLAP with DM was carried out by Han [8] who proposed a theoretical framework for defining OLAP mining functions. His *mining then cubing* function enables application of OLAP operators on the mining results. An example of implementing such function can be found in the Microsoft SQL Server and is denoted *data mining dimensions* [18]. These dimensions contain classifications obtained via clustering or other algorithms on the original facts and can be materialized and used (with some limitations) just like ordinary OLAP dimensions. Usman et al. [30] review the research literature on coupling OLAP and DM and propose a conceptual model for combining enhanced OLAP with data mining systems. The urge to enhance the analysis by integrating OLAP and DM was expressed in multiple publications in the past. Significant works in this area include [9], [31], [5], and [4]. The concept of Online Analytical Mining (OLAM) as the integration of OLAP and DM was introduced by Han et al. [9].

Extending the limitations of the multidimensional data model is another actively researched subject in theory and practice. A decade ago Pedersen et al. [22] formulated 11 requirements of comprehensive data analysis, evaluated 14 state-of-the-art data models for data warehousing against those requirements, and proposed an extended model for capturing and querying complex multidimensional data. A similar attempt to classify and evaluate existing multidimensional models is presented in [1]. However, the authors defined two orthogonal sets of classification criteria, namely, according to the kind of constructs/concepts they provide and according to the design phase at which they are employed. Another assessment of conceptual models is provided in [17], in which 6 prominent multidimensional models are evaluated against an exhaustive set of requirements regarding facts, dimensions, measures, operators etc. A survey of research achievements on providing OLAP to complex data can be found in [19].

A spectacular area of providing OLAP is that of the social media analysis. Rapid expansion and extreme popularity of social networking have confronted the underlying backend architectures with unprecedented volumes of user-generated content. Thusoo et al. from the Facebook developer team describe the challenges of implementing a DW for data-intensive Facebook applications and present a number of contributed open source technologies for warehousing petabytes of data in [28]. Twitter is another leading social network with acute demand for a data warehouse solution. The first quantitative study on Twitter was published in 2010 by Kwak et al. [16] who investigated Twitter's topological characteristics and its power as a new medium of information sharing. The authors obtained the data for their study by crawling the entire Twitter site as no API was available at that time. Twitter API framework launched in 2009 inspired thousands of application development projects including a number of research initiatives. Industrial applications are mostly marketing oriented, while other Twitter analysis works focus on improving the search and navigation in a huge flow of messages as well as on discovering valuable information about the contents and the users. We are more interested in the latter types of works as we pursue a multi-purpose analysis approach.

In 2007 Java et al. [12] presented their observations of the microblogging phenomena by studying the topological and geographical properties of Twitter's social network. They came up with a few categories for Twitter usage, such as daily chatter, information and URL sharing or news reporting. Mathioudakis and Koudas [20] proposed a tool called Twitter Monitor for detecting trends from Twitter streams in real-time by identifying emerging topics and bursty keywords. Recommendation systems for Twitter messages are presented by Chen et al. [3] and Phelan et al. [23]. Chen et al. studied content recommendation on Twitter to better direct user attention. Phelan et al. also considered RSS feeds as another source for information extraction to discover Twitter messages best matching the user's needs. Michelson and Macskassy [21] discover main topics of interest of Twitter users from the entities mentioned in their tweets. Hecht et al. [10] analyze unstructured information in the user profile's location field for location-based user categorization.

Recent explosion of Twitter-related research confirms the recognized potential for knowledge discovery from its data. In this work we exploit the advantages of the established OLAP technology coupled with DM to enable aggregation-centric analysis of the meta-data about the Twitter users and their messaging activity.

2. ACQUIRING FACTS AND DIMENSIONS

To explicify the challenges of transforming semi-structured data into multidimensional cubes, let us recall the relevant concepts of the data warehouse design. Data in a data warehouse is structured according to the aggregation-centric multidimensional data model that uses numeric measures as its analysis objects [2]. A *fact* entry represents the finest level of detail and normally corresponds to a single transaction or event occurrence. A fact consists of one or multiple measures, such as performance indicators, along with their descriptive properties referred to as *dimensions*. Values in a dimension can be structured into a *hierarchy* of granularity levels to enable drill-down and roll-up operations. Natural representation of a set of facts with their associated dimensions and classification hierarchies is a *multidimensional data cube*. Dimensions in a cube represent orthogonal characteristics of its measure(s). Each dimension is an axis in a multidimensional space with its *member* values as coordinates. Finally, each cell contains a value of the measure defined by the respective coordinates.

The terms *fact* and *measure* are often used as synonyms in the DW context. In our work, it appears crucial to distinguish between those terms to account for facts without measures. According to Kimball [13], a fact is given by a many-to-many relationship between a set of attributes. Some scenarios require storing many-to-many mappings in which no attribute qualifies as a measure. Typical cases are event records, where an event is given by a combination of simultaneously occurring dimensional characteristics. Kimball proposed to refer to such scenarios as *factless fact tables*[13]. Mansmann [19] suggests to use a more implementation-independent and less controversial term *non-measurable fact type*.

Another relevant term is that of *Slowly Changing Dimensions* (SCD) introduced by Kimball [13] and formally summarized in [26]. Classically, dimensions in a data cube correspond to non-volatile characteristics of the data. In reality, however, the instance or even the structure of a dimension may be subject to changes. The problem of SCD is well elaborated in the literature, with various strategies proposed for maintaining either the up-to-date or the historical view, or even the entire history of the evolution. Most strategies employ some kind of multi-versioning to preserve various states of the aggregates. Saddat et al. [26] describe a methodology for multi-version querying in the presence of SCD.

2.1 Data Transformation

Mapping semi-structured data to multidimensional cubes is generally a challenging task since the original format admits heterogeneity while the target one enforces a rigid structure. In case of the Twitter stream, the degree of heterogeneity is rather low and affects only a few data fields. We investigated the structure of the streamed data by converting JSON objects into an XML and buffering the output into a native XML database BaseX [11] developed within our working group. The following XML snippet gives an example of a converted tweet object:

```
<tweet>
  <text>
    Earthquake with the.scale of 8.9 magnitude
    #PrayForIndonesia #PrayForSumatera
  </text>
  <date>Wed Apr 11 08:57:02 +0000 2012</date>
  <source>web</source>
  <retweeted>false</retweeted>
  <user>
    <name>Miley ***</name>
    <date>Tue Jun 22 08:33:12 +0000 2010</date>
    <statuses_count>13101</statuses_count>
    <followers_count>1019</followers_count>
  </user>
</tweet>
```

We use the highly efficient BaseX storage [7] as a staging area where the data undergoes further transformations to prepare its loading into the DW. The entire transformation process consists of obtaining the relational view from the XML schema and mapping the latter to the multidimensional model.

The intermediate step of obtaining a relational model of the data is helpful in identifying classes of objects, their attributes and relationships, appropriate value domains of the attributes, cardinalities of the relationships, and integrity constraints. Figure 1 shows the resulting relational representation in the UML notation.

The subsequent step of obtaining a multidimensional perspective of the same data is performed in a semi-automated fashion. The manual part is concerned with semantic interpretation of the data and specifying the facts and the measures of interest as well as desired dimensions of the analysis. The automated part is a cardinality-based definition of facts and dimensions as described in [19]. The data model of Twitter contains only a small set of numeric attributes, which can be treated as measures. These attributes encompass the counters in the user profile and the tweet record. Other attributes are of descriptive nature and, therefore, should be mapped to dimensions or dimension hierarchies. Our approach is to treat a tweet event as a fact of the finest grain, with time, location, and user characteristics as its dimensions. All other characteristics are included into the respective dimensions or extracted into other facts.

A dimension is a *one-to-many* characteristic of a fact and can be of arbitrary complexity, from a single data field to a large collection of related attributes, from uniform granularity to a hierarchical structure with multiple alternative and parallel hierarchies. At the stage of the conceptual modeling, a dimension is structured as a graph of hierarchy levels as nodes and the "rolls-up-to" relationships between them as edges. We adopt the graphical notation of the extended Dimensional Fact Model (x-DFM) [19], which makes provisions for various kinds of behaviors in OLAP dimensions which is an extension of the Dimensional Fact Model (DFM) of Golfarelli et al. [6]. The x-DFM provides some advanced constructs, such as derived measures and categories, degenerated dimensions and fuzzy hierarchies, relevant for our model. Figure 2 shows a fragment of modeling a cube for storing various cumulative measures of the user activity in the x-DFM. The structure of the cube is a graph centered at the fact type node (*TweetCount*), which includes all measures (*#friends*, *#followers*, *#status*, *#favorited* and *#listed*) and a degenerated (i.e., consisting of a single data field) dimension (*FactID*). Dimensions are modeled as outgoing aggregation paths. All paths of a dimension converge in an abstract $\top$ node, which corresponds to the aggregated value *all*. A level node in a dimension consists of at least one key attribute, but may include further attributes represented as underlined terminal nodes.

2.2 Discovering new elements

So far, we only considered the explicitly available characteristics of the original set for constructing the cube. Once those characteristics have been mapped, the resulting model can be refined and enriched by adding new elements of type measure, category, dimension or even an entire cube. We were able refine the original dataset and its multidimensional view by applying the following techniques:

1. **Use of additional data sources and APIs**. Inclusion of external sources provides an opportunity to add new dimensional characteristics to a datacube. Here are some prominent examples applicable to the Twitter data. A useful property of the tweet's language can be added by using a language detection API, such as the one offered by Google or JSON. Another important property to detect is whether a tweet is spam or has malicious content. This can be done by employing the APIs of Askimed and Defensio or another similar service. We are currently working on integrating the above language and spam detection features to enable comprehensive text analysis of user-generated content. Twitter's own Search API can be used to retrieve daily trending topics and enrich tweet records with topic assignment. Currently we are working on enabling event and entity detection as well as adding sentiment information to the message's content.

2. **Derivation from existing characteristics**. Dimensions of a datacube are expected to be orthogonal, i.e., unrelated to one another. In practice, however, it may be beneficial to derive new characteristics from the existing ones and materialize their instances to be able to aggregate along them in OLAP queries. For example, one could add a new tweet dimension *media type* with values *"plain text"*, *"image"*, *"video"*, etc. based on the embedded multimedia content in the tweet message.

3. **Use of knowledge discovery techniques**. DM algorithms are helpful for discovering less obvious or hidden relationships in the dataset. Identified clusters or associations can be used as grouping criteria just like statically defined dimension categories.

Note that each of the added characteristics can serve as an input for discovering new characteristics, alone of in combination with other properties. For instance, identifying the language of the content and generating a machine translation of the text into a common default language by using the Google API opens up an opportunity to create a multilingual hierarchy of topics, keywords, hashtags, etc. and thus enable a cross-lingual aggregation.

In the next section we concentrate on the process of defining and maintaining discovered elements as well as their usage in OLAP queries.

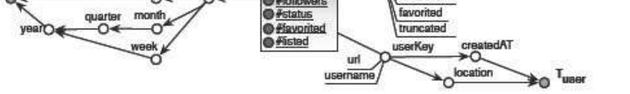


Figure 2: Tweet fact in the x-DFM

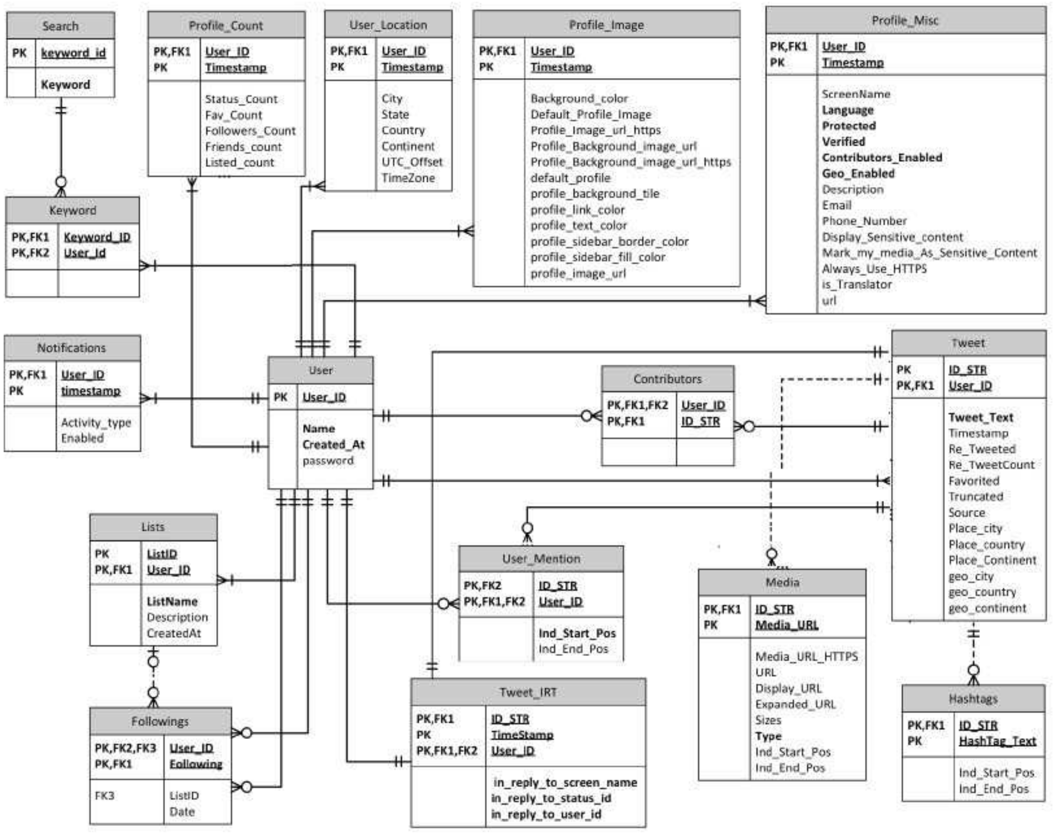


Figure 1: Relational view of the Twitter stream as a UML Class Diagram

3. MODELING DISCOVERED ELEMENTS

In general, a datacube can be extended by adding new elements of type *measure* or *dimension category*. A measure is a simple atomic field of a fact entry. Therefore, computing a new field of this type does not require additional adjustments to the overall cube structure. However, adding a new dimension or a hierarchy level to an existing dimension imposes a number of challenges with respect to modeling, implementing, querying, and maintaining such added element. We demonstrate the differences in handling static, derived, and discovered dimension categories at the example of the *user* dimension in *TweetCount* cube depicted in Figure 2, with its bottom-level category *userKey* with parallel roll-ups by creation date and by location.

We introduce an additional derived hierarchy *ranking* → *ranking group* → *popularity* based on the user's ranking in terms of the number of that user's followers and friends. There exist different methods of computing the ranking of the Twitter user, but most of them agree on the prevailing role of the number of followers. We adopt a simple formula

$$\text{ranking} = 0.8 * \#\text{followers} + 0.2 * \#\text{friends}$$

where *#friends* and *#followers* are the user's most recent measures from the cube *TweetCount*. With the proposed computation, the ranking values may vary from 0 to about 25 Mln. Therefore, it appears feasible to introduce additional levels of grain for this property. We adopt a percentage based roll-up into *ranking group* with 100 instances, one for each per cent of the total number of users. The last hierarchy level is *popularity* with just five instances, such as *superstar*, *celebrity*, *popular*, *regular*, and *unpopular*, with percentage based assignment (e.g., the bottom 20% are considered *unpopular*, the top 5% are *superstar*, etc.). Figure 3 shows the schema of the proposed dimension (left) as well as a fragment of its hierarchy instance (right). Obviously, the instance of such a computed hierarchy reflects the state of the data valid at the moment of its computation. The validity of this assignment becomes obsolete once the underlying fields *#friends* and *#followers* get modified.

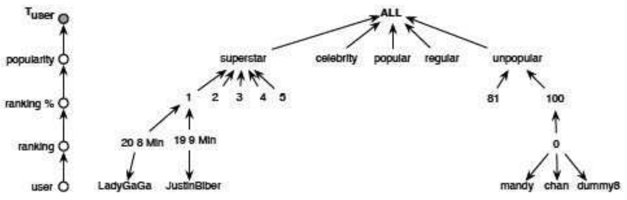


Figure 3: User rating dimension

Another example of an interesting dynamic classification in the user dimension is a taxonomy of user intentions introduced in [12]. The instances of *user intension* comprise *Daily Chatter*, *Conversations*, *Sharing Information*, and *Reporting News*. Assignment of a user to one of the instances in this classification is based on the analysis of multiple criteria including the frequency of twitting, writing direct responses to other users, linking to other sources, focusing on specific

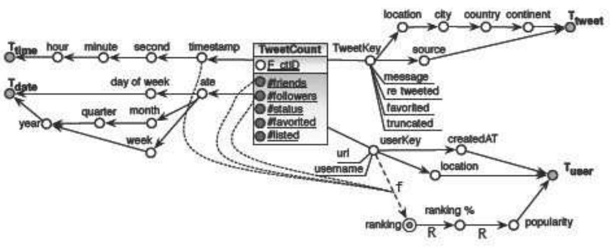


Figure 4: Adding ranking to the user dimension

topics, etc. Since many of the users may display multiple intension patterns and the intension of a user can evolve over time, the primary current intension can be determined with the help of data mining methods, such as clustering.

Both classifications, i.e., *ranking* and *user intension*, introduced here have a common property distinguishing them from standard OLAP dimensions, namely, their sensitivity to the evolution of the underlying dataset. In the extreme case of requiring full consistency with the current state, that would mean the necessity to update the dynamic elements after each insertion of new fact entries to the cube. That leads us to a more general problem of coping with changes in dimensions to be discussed in the next chapter.

Back to the task of the conceptual modeling of dynamic elements, obviously, the formal and the graphical notation of the multidimensional data model have to be extended to support such elements. The extended Dimensional Fact Model (x-DFM) [19] provides graphical elements for specifying derived categories. We adjust this notation to specify how a dynamic element is computed. Figure 4 shows the results of adding the user ranking hierarchy to the original cube. The derived category ranking is added as a parent level of userKey and is linked to the elements it is computed from by dotted lines. Since the ranking is computed from the most recent number of followers and friends, the linked input fields are the measures *#followers* and *#friends* as well as the timestamp category of the time dimension. The label "f" attached to the roll-up edge specifies that the category is computed based on a formula. Roll-up from ranking to ranking % and popularity is a rule-based one (label "R") and does not involve any extra input fields. In a similar fashion, roll-up edge notation can be extended to specify characteristics extracted with the help of external services, APIs, etc.

Whenever DM algorithms are used for creating a discovered category, such as the *user intension* in our example, the available derivation notation may be insufficient. In [24] we presented an approach to modeling mined dimensions based on symmetric treatment of measures and dimensions for obtaining a homogeneous graph of all data fields and hierarchical relationships between them.

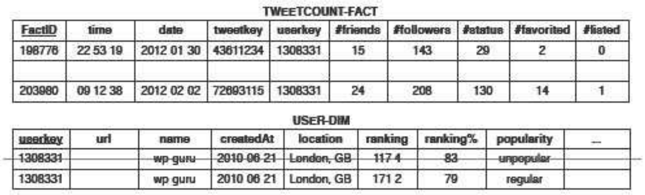
4. MAINTAINING DYNAMIC ELEMENTS

Discovered elements of type dimension category may be of a static nature (e.g., language, sentiment, topic) or evolve over time along with the evolution of the dataset. The former type can be treated just as a full-fledged dimension category since it does not impose any additional constraints on maintaining the data. The latter type, however, behaves similarly to a *changing dimension* – a term introduced by Kimball in [14]. Kimball distinguishes between slowly and rapidly changing dimensions and identifies various patterns of change occurrence. Several strategies of handling changes in OLAP dimensions have been proposed in the literature and implemented in leading data warehouse systems. Even though none of the previously identified evolution patterns and implementation alternatives deals with the dynamics of discovered categories proposed in this work, we wish to investigate to which extend the former can be adopted for such scenarios.

Let us recall various types of responding to change according to Kimball and apply them to our examples.

Type 0 response is a passive approach in which no action is taken to reflect the changes in the dimension. A single instance of the dimension exists, in which all attributes preserve their original values. This option of preserving only the historical view may appear satisfactory for some scenarios, but inadmissible in the general case. With dynamic categories, it is obviously necessary to keep the track of the changes in such categories for an up-to-date assignment.

Type 1 response to SCD is to simply overwrite old values with new ones. With this option, a single instance of the dimension is being maintained, in which all values correspond to the most recent assignment. Applying this option to storing the user's rating and intension values for Twitter analysis would mean inevitable loss of all previously computed values of these characteristics. Consequently, there will be no possibility to analyze the evolution of those characteristics or to perform historically correct aggregation. Figure 5 illustrates the effects of storing the user rating according to Type 1.



TWEETCOUNT-FACT

FactID	time	date	tweetkey	userkey	#friends	#followers	#status	#favorited	#listed
198776	22 53 19	2012 01 30	43611234	1308331	15	143	29	2	0
203980	09 12 38	2012 02 02	72693115	1308331	24	208	130	14	1

USER-DIM

userkey	url	name	createdAt	location	ranking	ranking%	popularity	...
~~1308331~~		~~wp-guru~~	~~2010 06 21~~	~~London, GB~~	~~117 4~~	~~83~~	~~unpopular~~	
1308331		wp guru	2010 06 21	London, GB	171 2	79	regular	

Figure 5: Type 1 SCD strategy for storing user ranking with no history preservation

Type 2 response aims at correct preservation of the prior history by adding a dimension row for each change. Since a single instance in a dimension is stored using multiple rows (one for each change), an extra surrogate key has to be introduced to uniquely identify each row and to be used as a foreign key from within the fact table. Figure 6 illustrates the effects of storing the user rating according to Type 2. A common extension of Type 2 storage is to add extra columns to the dimension table for storing the start and the end timestamp for each version. Even though this solution provides an accurate change tracking and ensures historically correct aggregation, it has a huge disadvantage of having to replace natural keys by the surrogate ones in the fact table. Especially with dynamic categories, whose values are computed from the fact entries, this approach would imply modification of the existing fact entries.

Type 3 method enables limited change tracking by using a separate column for each version of the changed attribute. This method is not an option for dynamic categories where we expect repeated and unlimited refreshment of the computed values in the dimensional table.

TWEETCOUNT FACT

FactID	time	date	tweetkey	userkey	#friends	#followers	#status	#favorited	#listed
198 6	22 53 19	2012 01 30	43611234	1308331a	15	143	29	2	0
203980	09 12 38	2012 02 02	2693115	1308331b	24	208	130	14	1

USER DIM

origkey	userkey	url	name	createdAt	location	ranking	ranking%	popularity	...
1308331	1308331a		wp guru	2010 06 21	London, GB	11 4	83	unpopular	
1308331	1308331b		wp guru	2010 06 21	London, GB	1 1 2	9	regular	

Figure 6: Type 2 SCD strategy for storing user ranking with history preservation

USER DIM

userkey	url	name	createdAt	location	ranking	ranking%	popularity	...
1308331		wp guru	2010 06 21	London, GB	1 1 2	9	regular	

USER HiSTORY

userkey	url	name	createdAt	location	ranking	ranking%	popularity		created
1308331		wp guru	2010 06 21	London, GB	11 4	83	unpopular		2012 02 01

Figure 7: Type 4 SCD strategy for storing user ranking with current and previous states

Type 4 response appears much more promising for managing multiple versions of the dimension's instance. This approach keeps the current data in the dimension table and extracts older versions into one or several '*'history tables"*. Figure 7 shows the storage of user dimension with only the current state in the dimension table and the previous states in the historical table.

Finally, **Type 6** is proposed as the hybrid of Types 1, 2, and 3. Just like Type 2, this solution also imposes the use of surrogate keys in the fact implementation.

Dynamic categories generated from the fact data through DM or other computations can be considered a special case of SCD, in which the changes occur with a certain regularity. The state of the dynamic category is guaranteed to be fully up-to-date, if it was computed from the most recent state of the underlying set of facts. However, it may be unaffordable to recompute the assignment each time new facts get inserted into the cube. Instead, interval-based or on-demand refreshment can be employed depending on the recency requirements and the prevailing change pattern. Back to our examples, user ranking is a rapidly evolving characteristic since the underlying counters of friends and followers change frequently at least for active users. As for user intension, this assignment is expected to be more stable as it is based on the prevailing usage patterns and clustering of similar behaviors.

Whatever refreshment strategy is used in a dimension with dynamic categories, Type 4 response to SCD has proven to offer an adequate solution for managing both the current version and all previous states of the dimension instance. No surrogate keys are necessary and no adjustments in the fact table implementation. The dimension's instance turns into a multi-versioned one, where a particular version can be retrieved by querying the timestamps of the instances.

Last not least, it appears crucial to normalize the dimension table according to the snowflake schema. In the existence of several dynamic categories or change patterns within a single dimension, storing all attributes and their assignments in the same dimensional table would lead to extreme redundancy and confusion. Decomposition into separated tables for each hierarchy level or at least each hierarchy path makes it possible to handle changes in that particular path using a dedicated history table.

4.1 OLAP Queries with multi-versioning

Adopting the Type 4 strategy to handling changes in the dimension generates a multi-versioned instance of any changing dimension. Availability of the current state as well as of each previously valid state makes it possible to perform historically correct aggregation by joining the fact entries with the matching versions of the dimension records. Besides, one can aggregate recent facts along a historical version of a dynamic characteristic or aggregate historical data along the current state of the changing category. Examples of queries containing a deliberate version mismatch are "retrieve the messages twitted in 2009 by the users who are popular now (and not in 2009!)", or "retrieve recent tweets containing the hashtags which were in top 20 in 2008".

If pre-aggregation is used for materializing the aggregates at different levels of grain, co-existence of multiple versions in a dimension does not cause problems because each fact entry has exactly one matching version of the dimension's record. Thereby, pre-aggregation produces historically correct values.

5. CONCLUSIONS

In this work we proposed to extract multidimensional datacubes for OLAP from semi-structured data sets and to extend the resulting model by including dynamic categories and hierarchies discovered from the data through DM methods and other computations. The discovered classifications reflect "hidden" relationships in the data set and thus represent new axes for exploring the measures in a cube.

As a non-conventional application for OLAP, we used the publicly available stream of the user-generated data provided by the Twitter platform. Tweeted messages streamed as semi-structured records o over 60 fields can be enriched with additionally extracted characteristics relevant for the analysis. We considered various sources of enriching the original set, from external services and APIs, to derivation from existing characteristics and application of knowledge discovery techniques.

We handled the process of adding discovered categories at the conceptual and logical level and investigated which approaches to implementing slowly changing dimensions are suitable for our scenario. The method of storing only the current state in the dimension table and extracting the previous versions into a history table proved to be the appropriate solution that ensures historically correct aggregation but also enables deliberate historically incorrect aggregation useful for investigating the data evolution itself.

Our approach was tested on the dataset of the Twitter's public stream with a focus on the metadata analysis. We presented examples of enriching the user dimension with computed and discovered properties. A multitude of other properties can be added to other dimensions in a similar fashion. In the future, we plan to extract additional properties from the content of the streamed messages, such as keywords, events and entities, sentiment, topics, embedded media, etc. to enable an in-depth analysis of the content generated within the given information chanel.

6. ACKNOWLEDGMENTS

This work was partially supported by DFG Research Training Group GK-1042 "Explorative Analysis and Visualization of Large Information Spaces", University of Konstanz.

7. REFERENCES

[1] A. Abelló, J. Samos, and F. Saltor. A framework for the classification and description of multidimensional data models. In *DEXA 2001: Proceedings of DEXA 2001*, pages 668–677. Springer-Verlag, 2001.

[2] S. Chaudhuri, U. Dayal, and V. Ganti. Database technology for decision support systems. *Computer*, 34(12):48–55, 2001.

[3] J. Chen, R. Nairn, L. Nelson, M. S. Bernstein, and E. H. Chi. Short and tweet: experiments on recommending content from information streams. In *Proc. CHI*, pages 1185–1194. ACM, 2010.

[4] F. Dehne, T. Eavis, and A. Rau-Chaplin. Coarse grained parallel on-line analytical processing (olap) for data mining. *Computational Science-ICCS 2001*, pages 589–598, 2001.

[5] S. Dzeroski, D. Hristovski, and B. Peterlin. Using data mining and olap to discover patterns in a database of patients with y-chromosome deletions. In *Proceedings of the AMIA Symposium*, page 215. American Medical Informatics Association, 2000.

[6] M. Golfarelli, D. Maio, and S. Rizzi. The Dimensional Fact Model: A conceptual model for data warehouses. *International Journal of Cooperative Information Systems*, 7(2-3):215–2–47, 1998.

[7] C. Grün, A. Holupirek, M. Kramis, M. H. Scholl, and M. Waldvogel. Pushing xpath accelerator to its limits. In *Proceedings of ExpDB*. ACM, 2006.

[8] J. Han. Olap mining: An integration of olap with data mining. In *Proc. of the 7th IFIP 2.6 Working Conf. on Database Semantics (DS-7*, 1997.

[9] J. Han, S. Chee, and J. Chiang. Issues for on-line analytical mining of data warehouses. In *Proc. of the Workshop on Research Issues on Data Mining and Knowledge Discovery, Seattle, Washington*, pages 2:1–2:5, 1998.

[10] B. Hecht, L. Hong, B. Suh, and E. H. Chi. Tweets from justin bieber's heart: the dynamics of the location field in user profiles. In *Proc. CHI*, pages 237–246, 2011.

[11] A. Holupirek, C. Grün, and M. H. Scholl. BaseX & DeepFS joint storage for filesystem and database. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, pages 1108–1111. ACM, 2009.

[12] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. ACM, 2007.

[13] R. Kimball. *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. John Wiley & Sons, Inc., New York, NY, USA, 1996.

[14] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[15] R. Krikorian. Developing for @twitterapi (techcrunch disrupt hackathon).

[16] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

[17] S. LujĞn-Mora, J. Trujillo, and I.-Y. Song. A uml profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering*, 59:725–769, December 2006.

[18] J. MacLennan, Z. Tang, and B. Crivat. *Mining OLAP Cubes*, chapter 13, pages 429–431. Wiley Publishing, 2008.

[19] S. Mansmann. *Extending the OLAP Technology to Handle Non-Conventional and Complex Data*. PhD thesis, Konstanz, Germany, 2008.

[20] M. Mathioudakis and N. Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 international conference on Management of data*, pages 1155–1158. ACM, 2010.

[21] M. Michelson and S. A. Macskassy. Discovering users' topics of interest on twitter: a first look. In *Proceedings of the Fourth Workshop on Analytics for Noisy Unstructured Text Data, AND 2010, Toronto Ontario, Canada, October 26th, 2010 (in conjunction with CIKM 2010)*. ACM, 2010.

[22] T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. A foundation for capturing and querying complex multidimensional data. *Information Systems*, 26(5):383–423, 2001.

[23] O. Phelan, K. McCarthy, and B. Smyth. Using twitter to recommend real-time topical news. In *Proceedings of the third ACM conference on Recommender systems*, pages 385–388. ACM, 2009.

[24] N. U. Rehman, S. Mansmann, A. Weiler, and M. H. Scholl. Building a data warehouse for twitter stream exploration. In *Proc. of 1st IEEE ACM International Workshop on Multi-agent Systems and Social Networks(MASSN2012), in conjunction with ASONAM 2012*, august 2012.

[25] K. Roebuck. *Big Data: High-Impact Strategies - What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors*. Emereo Pty Limited, 2011.

[26] E. Saddad, A. El-Bastawissy, M. Rafea, and O. Hegazy. Multiversion queries in multidimensional structures. In *Proceedings of the 6th International Conference on Informatics and Systems (INFOS08)*, pages DB42–DB50, march 2008.

[27] C. Strauch. Nosql databases. lecture selected topics on software-technology ultra-large scale sites, manuscript. *Lecture Notes, Stuttgart Media University*, 2011.

[28] A. Thusoo, Z. Shao, S. Anthony, D. Borthakur, N. Jain, J. Sen Sarma, R. Murthy, and H. Liu. Data warehousing and analytics infrastructure at facebook. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10, pages 1013–1020, New York, NY, USA, 2010. ACM.

[29] Twitter Team. Twitter turns six, 2012.

[30] M. Usman, S. Asghar, and S. Fong. A conceptual model for combining enhanced olap and data mining systems. In *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, pages 1958–1963. IEEE, 2009.

[31] H. Zhu. *On-line analytical mining of association rules*. PhD thesis, Simon Fraser University, 1998.