# Semantics and Usage Statistics for Multi-Dimensional Query Expansion

Raphaël Thollot, Nicolas Kuchmann-Beauger, Marie-Aude Aufaure

# Semantics and Usage Statistics for Multi-Dimensional Query Expansion

Raphaël Thollot[1], Nicolas Kuchmann-Beauger[1], and Marie-Aude Aufaure[2]

[1] SAP Research - BI practice, 157 rue Anatole France, 92309 Levallois-Perret, France,
raphael.thollot@sap.com, nicolas.kuchmann-beauger@sap.com,
[2] Ecole Centrale Paris, MAS laboratory, 92290 Chatenay-Malabry, France,
marie-aude.aufaure@ecp.fr

**Abstract.** As the amount and complexity of data keeps increasing in data warehouses, their exploration for analytical purposes may be hindered. Recommender systems have grown very popular on the Web with sites like Amazon, Netflix, etc. These systems proved successful to help users explore available content related to what they are currently looking at. Recent systems consider the use of recommendation techniques to suggest data warehouse queries and help an analyst pursue its exploration. In this paper, we present a personalized query expansion component which suggests measures and dimensions to iteratively build consistent queries over a data warehouse. Our approach leverages (a) semantics defined in multi-dimensional domain models, (b) collaborative usage statistics derived from existing repositories of Business Intelligence documents like dashboards and reports and (c) preferences defined in a user profile. We finally present results obtained with a prototype implementation of an interactive query designer.

**Keywords:** query, expansion, recommendation, multi-dimensional, OLAP

## 1 Introduction

Data warehouses (DW) are designed to integrate and prepare data from production systems to be analyzed with Business Intelligence (BI) tools. End-users can navigate through and analyze large amounts of data thanks to a significant effort from IT and domain experts to first model domains of interests. However, exploiting these multi-dimensional models may become challenging in important deployments of production systems. Indeed, they can grow extremely complex with thousands of BI entities, measures and dimensions used, e.g., to build OLAP cubes.

In common reporting and analysis tools, end-users can design data queries using some kind of query panel. For instance, a user may drag and drop measures and dimensions she wants to use to create a new visualization, e.g., showing the `Sales revenue` aggregated by `City`. Given the number of available measures and dimensions, it is crucial to help the user iteratively build her query. We

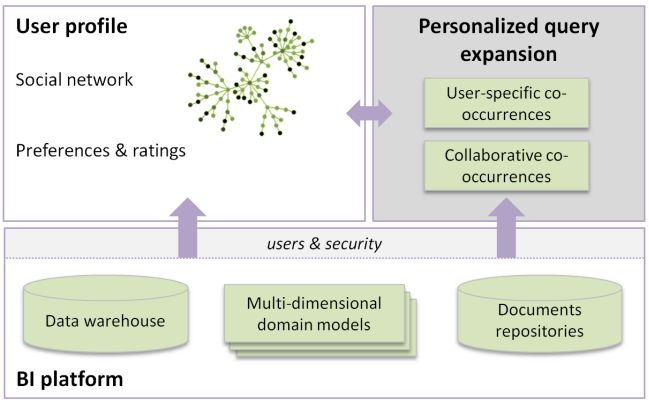


**Fig. 1.** Architecture overview of the proposed personalized query expansion system for multi-dimensional models.

define the iterative query expansion problem as a function $QE$ taking as input a user $u$, the current query $q$ and additional parameters $params$. This function returns a collection of scored queries $(q_i, r_i)$ such that, for all $i$ from 1 to $n$, $|q_i| = |q| + 1$ and $q \subset q_i$:

$$QE\colon (u, q, params) \mapsto \{(q_1, r_1), \ldots, (q_n, r_n)\}$$

The problem is thus to find candidate measures and dimensions that are best associated with $q$. In response to this, this paper presents our solution which was experimented by building an interactive and personalized query designer. Our method leverages semantics of multi-dimensional models, collaborative usage statistics derived from repositories of BI documents and user preferences to iteratively suggest relevant measures and dimensions. Figure 1 illustrates the main components involved in the architecture of our system.

In the rest of this article, Section 2 introduces multi-dimensional domain models and their semantics. Section 3 presents a collaborative measure of co-occurrence between entities of these models. Then, Section 4 introduces users' preferences and our personalized query expansion component. Section 5 describes the system architecture and presents results obtained with the implementation of an interactive query designer. Finally, we review related work in Section 6.

## 2 Semantics of multi-dimensional domain models

Multi-dimensional models for DWs define concepts of the business domain with key indicators (*measures*) and axis of analysis (*dimensions*).

### 2.1 Measures and dimensions

Measures are numerical facts that can be aggregated against various dimensions [3]. For instance, the measure `Sales revenue` could be aggregated (e.g.,

from unit sales) on dimensions `Country` and `Product` to get the revenue generated by products sold in different countries.

Business domain models are used to query the DW for actual data and perform calculations. A DW may be materialized as a relational database, and queries thus have to be expressed accordingly, for instance as SQL. From a calculation point of view, it is also possible to build multi-dimensional OLAP cubes. Measures are aggregated inside the different cells of the cube formed by dimensions. Queries can be expressed on these cubes, e.g., with Multi-Dimensional eXpressions (MDX). Beyond this, modern DWs provide SQL/MDX generation algorithms to enable non-expert users to formulate ad-hoc queries.

In the next section, we present hierarchies and functional dependencies (between measures and dimensions) that multi-dimensional domain models may also define.

### 2.2 Functional dependencies and hierarchies

Two objects (measures or dimensions) are functionally dependant if one *determines* the other. For instance, knowing the `City` determines the related `State`. Another example that involves a measure and a dimension is to say that the `Sales revenue` is determined by a `Customer` (e.g., aggregated from unit sales in a fact table). Functional dependencies are transitive: if `A` determines `B` which determines `C`, then `A` determines `C`. In the most simple scenario, all measures are determined by all dimensions. This is the case when using a basic dataset, for instance reduced to one fact table with dimensions in a *star schema*.

Functional dependencies are important to compose meaningful queries. For instance, they can be used to ensure suggested queries do not contain incompatible objects which would prevent their execution. However, business domain models do not necessarily capture and expose this information. Hierarchies of dimensions are more common though, usually exploited in reporting and analysis tools to enable the *drill-down* operation. For instance, if a `Year - Quarter` hierarchy is defined, the result of a user drilling down on `Year` 2010 is a more fine-grained query with the `Quarter` dimension, filtered on `Year` = 2010. If hierarchies of dimensions can be used to determine minimal dependency chains, techniques are required to help with automatic detection of functional dependencies. In particular, the approach presented by [12] is to create *DL-Lite* domain ontologies from conceptual schemas and use inferencing capabilities.

## 3 Usage statistics in BI documents

Functional dependencies and hierarchies previously presented provide very structural knowledge regarding associations between BI entities. Beyond this, some BI platforms propose repositories of documents like reports or dashboards which can be used to compute actual usage statistics for measures and dimensions. This kind of information is extremely valuable in our use case, since query expansion implies to find the best candidate to associate to a given set of measures and dimensions.

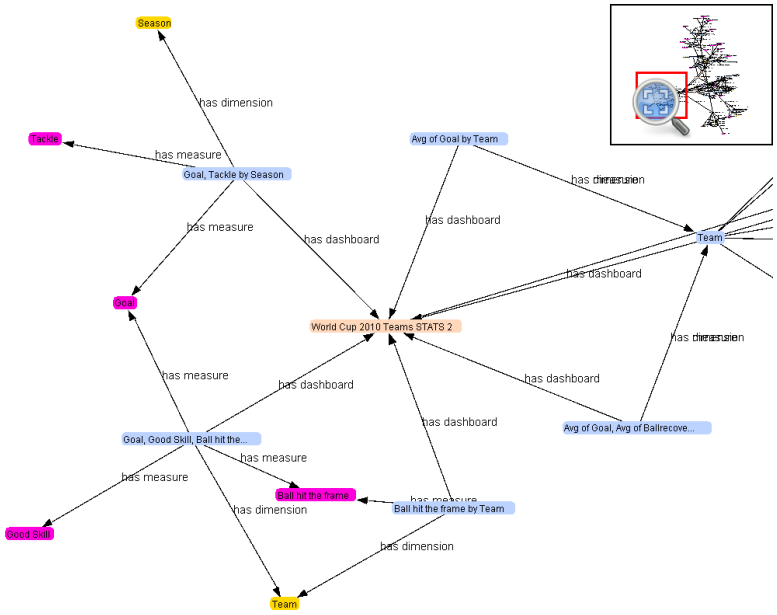


**Fig. 2.** Graph describing a dashboard (orange) and associated charts (blue), with referenced measures (purple) and dimensions (yellow).

### 3.1 Structure of BI documents and co-occurrence

We use the structure of BI documents to define co-occurrences between measures and dimensions. For instance, BI reports are roughly composed of sections which may contain charts, tables, text areas for comments, etc. Charts and tables define important units of sense. Measures and dimensions associated in a same table/chart are likely to be strongly related and represent an analysis of specific interest to the user. Similarly, dashboards can be composed of different pages or views which contain charts and tables. Figure 2 illustrates the graph representation of the dashboard *World Cup Team STATS 2* and its associated charts. More generally, any BI document referencing measures and dimensions could be used to derive consolidated co-occurrences or usage statistics.

### 3.2 Personal co-occurrence measure

BI platforms provide access control rules for business domain models and documents built on top of them. Consequently, different users may not have access to the same models and at a more fine-grained level to the same measures and dimensions. Besides, repositories contain documents generated by and shared (or not) between different users of the system. As a result, the measure of co-occurrence that we define in this section is inherently personalized. Let us consider a user $u$ and let $occ_u(e_1)$ denote the set of charts and tables – visible to

the user $u$ – referencing a BI entity $e_1$, measure or dimension. We define the co-occurrence of two entities $e_1$ and $e_2$ as the Jaccard index of the sets $occ_u(e_1)$ and $occ_u(e_2)$:

$$cooc_u(e_1, e_2) = J(occ_u(e_1), occ_u(e_2)) = \frac{|occ_u(e_1) \cap occ_u(e_2)|}{|occ_u(e_1) \cup occ_u(e_2)|} \tag{1}$$

The Jaccard index is a simple but commonly used measure of the similarity between two sample sets.

### 3.3 Collaborative co-occurrence measure

**Cold-start users and coverage.** In recommender systems (RS), the *coverage* is the percentage of items that can actually be recommended, similar to the recall in information retrieval. Formula 1 presents a problem for *cold-start* users, i.e. those new to the system. Indeed, these users do not have stored documents from which co-occurrences can be computed. Collaborative RS introduce the contribution of other users in the item scoring function to improve the system's coverage and enable the exploration of resources previously unknowned (or unused) by the user. A simple approach consists in using a linear combination of the user-specific value and the average over the set of all *users*.

**Using the social/trust network.** The simple approach previously described broadens the collaborative contribution to "the whole world" and all users have the same weight. Trust-based RS have illustrated the importance of considering the user's social network and, e.g., favoring users close to the current user [7]. Narrowing the collaborative contribution down to close users presents benefits at two levels: (a) results are more precisely personalized and (b) potential precomputation is reduced.

Let us note $SN(u)$ the set of users in $u$'s social network which can be filtered, e.g., to keep only users up to a certain maximum distance. We propose the following refined co-occurrence measure, were $\alpha$ and $\beta$ are positive coefficients to be adjusted experimentally such that $\alpha + \beta = 1$:

$$\begin{aligned} cooc(u, e_1, e_2) = \alpha \cdot cooc_u(e_1, e_2) \\ + \frac{\beta}{|SN(u)|} \cdot \sum_{u' \in SN(u)} \frac{1}{d(u, u')} cooc_{u'}(e_1, e_2) \end{aligned} \tag{2}$$

This measure $cooc(u, e_1, e_2)$ is defined for entities $e_1$ and $e_2$ exposed to the user $u$ by access control rules. The contribution of each user $u'$ is weighted by the inverse of the distance $d(u, u')$.

Relations between users can be obtained from a variety of sources, including popular social networks on the Web. However, this does not necessarily match corporate requirements since users of the system are actual employees of a same company. In this context, enterprise directories can be used to extract, e.g.,

hierarchical relations between employees. Clearly, other types of relations may be considered but the actual construction of the social network is beyond the scope of this paper.

## 4 Personalized query expansion

In this section, we describe our approach to design a personalized query expansion component leveraging models semantics, co-occurrences and user preferences.

### 4.1 User preferences

We distinguish *explicit* and *implicit* preferences, respectively noted $pref_{u,expl}$ and $pref_{u,impl}$. For a given entity $e$, we define the user's preference function $pref_u$ as a linear combination of both preferences, for instance simply:

$$pref_u(e) = \frac{1}{2}\left(pref_{u,impl}(e) + pref_{u,expl}(e)\right) \tag{3}$$

Explicit preferences are feedback received from the user, e.g., in the form of ratings (in $[0, 1]$) assigned to measures, dimensions. Let us note $r_{u,e}$ the rating given by $u$ to $e$ and $\overline{r_u}$ the average rating given by $u$. We define $pref_{u,expl}(e) = r_{u,e}$ if $u$ has already rated $e$, and $pref_{u,expl}(e) = \overline{r_u}$ otherwise.

Implicit preferences can be derived from a variety of sources, for instance by analyzing logs of queries executed in users' sessions [4]. In our case, we consider occurrences of BI entities in documents manipulated by the user as a simple indicator of such preferences:

$$pref_{u,impl}(e) = \frac{|occ_u(e)|}{\max_{e'}(|occ_u(e')|)} \tag{4}$$

### 4.2 Query expansion

The aim of our system is to assist the user in the query design phase by offering suggestions of measures and dimensions she could use to explore data. When she selects a measure or a dimension, it is added to the query being built and suggestions are refreshed to form new consistently augmented queries.

**Ranking.** To complete a given query $q = \{e_1, \ldots, e_n\}$ with an additional measure or dimension, we need to find candidate entities and rank them. Candidate entities, $c_j, j = 1..p$, are those defined in the same domain and compatible with every $e_i$, determined using functional dependencies (see Section 2.2). We then use the following personalized function to rank each candidate $c_j$:

$$rank_u(c_j, q) = \begin{cases} pref_u(c_j) & \text{if } q = \emptyset \\ pref_u(c_j) \cdot \frac{1}{n} \sum_{i=1}^{n} cooc(u, c_j, e_i) & \text{otherwise} \end{cases} \tag{5}$$

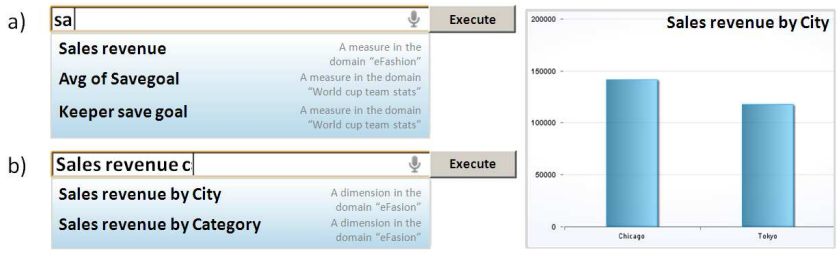


**Fig. 3.** Screenshot of auto-completion used in an interactive query designer. (a) First suggestions after characters "sa" and (b) suggestions following the selection of measure `Sales revenue` and character "c". On the right is a sample visualization that can be built with the query `Sales revenue by City`.

To conclude with the notation of the query expansion problem introduced in Section 1, we define our component $QE$ as:

$$QE\colon (u, q, params) \mapsto \{(q_1, rank_u(c_1, q)), \ldots, (q_p, rank_u(c_p, q))\}$$

**Parameters.** Beyond ranking, suggestions of the query expansion component can be fine-tuned using various parameters:

- The maximum number of results.
- The type of suggested entities can be limited to measures and/or dimensions.
- The domain can be restricted to a list of accepted models.
- Suggested dimensions can be grouped by and limited to certain hierarchies. This may be used to reduce the number of suggestions and encourage the user explore varied axis of analysis.

## 5 Experimentation: auto-completion in a query designer

In previous sections we presented tools used to implement an interactive query designer. In this section, we illustrate results obtained with a prototype implementation. We developed a query designer which simply presents a search text box to the user. As she types, candidate measures and dimensions are proposed to the user as auto-completion suggestions.

Figure 3.a) shows measures (from distinct domain models) suggested when the user starts typing "sa": `Sales revenue`, `Avg of savegoal` and `Keeper save goal`. In Figure 3.b), the user has selected the first suggestion `Sales revenue` and keeps typing "c". The system suggests the two dimensions `City` and `Category`. The auto-completion initialization requires that the user roughly knows the names of objects she wants to manipulate, which may be a barrier to adoption. To help her get started and explore available data, suggestions can be surfaced to the user before she even starts typing. For instance, the most commonly used

| Measure | Dimension | Co-occurrence |
|---|---|---|
| Sales Revenue | Quarter | 0,38 |
| | State | 0,25 |
| | Year | 0,25 |
| | Category | 0,25 |
| | Lines | 0,22 |

**Table 1.** Top-5 dimensions that most co-occur (in a collection of dashboards) with the `Sales Revenue` measure.

measures and dimensions of various domain models could be suggested to start with.

In our implementation of the architecture presented in Figure 1, we relied on the BI platform *SAP BI 4*. Documents wise, this platform proposes reporting and dashboarding solutions respectively named *WebIntelligence* and *SAP BusinessObjects Explorer* (or *Exploration Views*). We experimented the computation of co-occurrences using dashboards accessible through the demonstration account of *SAP Exploration Views* on-demand[3]. This account exposes 9 dashboards which contain 31 charts. The 7 underlying domain models define 54 dimensions and 86 measures. Table 1 presents the 5 dimensions that most co-occur with a given measure named `Sales Revenue`. From the social network point of view, we build on the prototype *Social Network Analyzer*[4]. In particular, APIs of this prototype expose the user's social graph at a depth of 2.

## 6 Related work

In this section we briefly review previous work related to personalization and recommendation in multi-dimensional DWs. Various types of OLAP recommendations can be considered with *interactive assistance for query design*, *anticipatory recommendations* and *alternative results* [8]. The work presented in this paper best corresponds to the first type which, to the best of our knowledge, has not been investigated much by previous research.

Techniques employed for query recommendations in DWs have been thoroughly reviewed by Marcel et al. [10]. In particular, the authors provide a formal framework to express query recommendations and divide them in methods (a) based on user profiles, (b) using query logs, (c) based on expectations and (d) hybrid ones. The first ones include user profiles (e.g., preferences) in the recommendation process to maximize the user's interest for suggested queries [6, 9]. Interestingly, recommendations may integrate visualization-related constraints [2]. Second, methods based on query logs mainly address predictive recommendations of forthcoming queries and position analysis sessions as first-class citizens [4]. One approach is to model query logs using a probabilistic Markov model. Another is to use a distance metric between sessions to extract recommended

[3] http://exploration-views.ondemand.com
[4] http://sna-demo.ondemand.com/

queries from past sessions similar to the current one. Methods based on expectations aim at determining and guiding the user toward zones of a cube that present unexpected data, for instance by maximizing the entropy [13]. Finally, these approaches may be combined in various ways with hybrid methods [5].

Recommendations of multi-dimensional queries is a fairly recent topic. However, RS have become a popular research area thanks to successful commercial applications, e.g., on e-commerce Web sites. RS are commonly categorized in content-based (CB) and collaborative filtering (CF) approaches [1]. CF methods usually rely on $user \times item$ matrices to compute similarities between items and users based on ratings, preferences, etc. [14]. The main assumption behind such techniques is that users with similar rating schemes will react similarly to other items. Preferences can be either explicit (ratings) or implicit (e.g., derived from click-through data). Metrics used in CF techniques build for instance on vector-based cosine similarity and *Pearson* correlation. The collaborative contribution can be refined by considering the use of a trust network between users [7]. CF techniques present a certain number of issues dealing with matrix sparsity and cold-start users. Besides, they remain superficial since they lack a representation of the actual item content. On the other hand, CB methods use descriptions of items' features (like weighted keywords in a vector space model) and assume that the user will like items similar to those he liked in the past. Finally, hybrid methods often combine CB and CF approaches to overcome their respective drawbacks [11].

## 7 Conclusion and Future Work

In this paper we presented a personalized query expansion system that leverages (a) semantics of multi-dimensional domain models, (b) usage statistics derived from (co-)occurrences of *measures* and *dimensions* in repositories of BI documents and (c) user preferences. The system was experimented with a prototype of interactive query designer, assisting the user with auto-completion suggestions. This experimentation showed encouraging usability results.

However, we did not manage to obtain a joint dataset between deployments of *SAP Exploration Views* and *Social Network Analyzer*. Therefore, we would like to focus in future work on the generation of such a dataset to conduct user-satisfaction tests. In particular, it would be interesting to highlight and measure the benefits of the collaborative contribution introduced in formula 2.

We illustrated our approach with an interactive query designer. Beyond this, we are currently investigating other promising applications of the concepts presented in this paper. For instance, the search-like interface of Figure 3 could be extended. In particular, we are considering the retrieval of charts – from existing reports and dashboards – with similar data. Also, user preferences and (co-)occurrences may be used in a *question answering* system to help, e.g., with personalized query reformulation.

More generally, we reckon that recommendations in the context of DWs and BI platforms could benefit much further from techniques developed in the RS

area. However, taking into account the specific semantics of multi-dimensional models is also key to provide relevant structured analytics.

## References


1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. 17(6), 734–749 (2005)
2. Bellatreche, L., Giacometti, A., Marcel, P., Mouloudi, H., Laurent, D.: A personalization framework for olap queries. In: Proceedings of the 8th ACM international workshop on Data warehousing and OLAP. pp. 9–18. DOLAP '05, ACM, New York, NY, USA (2005)
3. Bhide, M., Chakravarthy, V., Gupta, A., Gupta, H., Mohania, M.K., Puniyani, K., Roy, P., Roy, S., Sengar, V.S.: Enhanced business intelligence using erocs. In: ICDE. pp. 1616–1619. IEEE (2008)
4. Giacometti, A., Marcel, P., Negre, E.: A framework for recommending olap queries. In: Proceeding of the ACM 11th international workshop on Data warehousing and OLAP. pp. 73–80. DOLAP '08, ACM, New York, NY, USA (2008)
5. Giacometti, A., Marcel, P., Negre, E., Soulet, A.: Query recommendations for olap discovery driven analysis. In: Proceeding of the 12th ACM workshop on Data warehousing and OLAP. pp. 81–88. DOLAP '09, ACM, New York, NY, USA (2009)
6. Golfarelli, M., Rizzi, S., Biondi, P.: myolap: An approach to express and evaluate olap preferences. IEEE Transactions on Knowledge and Data Engineering 23, 1050–1064 (2011)
7. Jamali, M., Ester, M.: *TrustWalker*: a random walk model for combining trust-based and item-based recommendation. In: IV, J.F.E., Fogelman-Soulié, F., Flach, P.A., Zaki, M.J. (eds.) KDD. pp. 397–406. ACM (2009)
8. Jerbi, H., Ravat, F., Teste, O., Zurfluh, G.: Preference-based recommendations for olap analysis. In: Pedersen, T., Mohania, M., Tjoa, A. (eds.) Data Warehousing and Knowledge Discovery, Lecture Notes in Computer Science, vol. 5691, pp. 467–478. Springer Berlin / Heidelberg (2009)
9. Kozmina, N., Niedrite, L.: Olap personalization with user-describing profiles. In: Forbrig, P., Gnther, H., Aalst, W., Mylopoulos, J., Sadeh, N.M., Shaw, M.J., Szyperski, C. (eds.) Perspectives in Business Informatics Research, Lecture Notes in Business Information Processing, vol. 64, pp. 188–202. Springer (2010)
10. Marcel, P., Negre, E.: A survey of query recommendation techniques for datawarehouse exploration. In: Proceedings of 7th Conference on Data Warehousing and On-Line Analysis (*Entrepts de Donnes et Analyse*), EDA'11 (2011)
11. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: in Eighteenth National Conference on Artificial Intelligence. pp. 187–192 (2002)
12. Romero, O., Calvanese, D., Abelló, A., Rodríguez-Muro, M.: Discovering functional dependencies for multidimensional design. In: Proceeding of the 12th ACM workshop on Data warehousing and OLAP. pp. 1–8. DOLAP '09, ACM, New York, USA (2009)
13. Sarawagi, S.: User-adaptive exploration of multidimensional data. In: Proceedings of the 26th Conference on Very Large DataBases (VLDB), Cairo, Egypt, 2000. (2000)
14. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence 2009, 4:2–4:2 (January 2009)