# Schema-Aware Recommendation Engine for SQL Queries

Vishnu Pratish
University of Waterloo
200 University Avenue West
Waterloo, Ontario, Canada
vpratis@uwaterloo.ca

Pragnya Addala
University of Waterloo
200 University Avenue West
Waterloo, Ontario, Canada
paddala@uwaterloo.ca

## ABSTRACT

This paper provides a sample of a LaTeX document which conforms to the formatting guidelines for ACM SIG Proceedings. It complements the document *Author's Guide to Preparing ACM SIG Proceedings Using LaTeX2$_\epsilon$ and BibTeX*. This source file has been written with the intention of being compiled under LaTeX2$_\epsilon$ and BibTeX.

The developers have tried to include every imaginable sort of "bells and whistles", such as a subtitle, footnotes on title, subtitle and authors, as well as in the text, and every optional component (e.g. Acknowledgments, Additional Authors, Appendices), not to mention examples of equations, theorems, tables and figures.

To make best use of this sample document, run it through LaTeX and BibTeX, and compare this source code with the printed output produced by the dvi file.

## Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

## General Terms

Theory

## Keywords

ACM proceedings, LaTeX, text tagging

## 1. INTRODUCTION

SQL is a universal language used by software engineers on a regular basis. Databases schemas are often designed by experts who are aware of the system and programmers often end up not having much familiarity with the schema. A software engineer who writes queries in a development environment based on this schema he will have to go back and forth to see the schema information to write effective queries. Doing this on a large scale system can be a tedious process. To explain this problem, we take the example of a user searching for the patient record of Englishman "William Scott" in an hospital database. Schema complexity poses as the first challenge. Most hospitals have records of their patients in varied schema, typical to each department. Second, the user may not be fully aware of the exact values of the selection predicates, and may give only a misspelled or partial attribute value (in this case, the user knows him as "Will"). Third, we would like the user to issue queries that are meaningful in terms of result size. A query listing all the patients in the hospital would not be helpful to the user, and in some cases could also be computationally expensive for the system. Lastly, we do not expect the user to be proficient in SQL to access the database.

Structured query models like XQuery, and the wider known SQL are the current existing means provided by database systems to allow users to express complex query semantics. Though powerful, these query models are in essence diïñČcult for users to adopt because they require users to fully understand the structure of the database (i.e., schema) and to express queries in terms of that particular structure. Most often, a developer due to inexperience, unfamiliarity with the database schema or mere lack of attention to computational complexity of the query, fails to use the optimal query. Recommendation engines for a SQL query have the potential to ease a programmer's life by providing him with possible suggestions and optimizations. Providing informed autosuggestions while writing the query can therefore ease a programmer's life. We use an informed approach towards this end by using the information in hand like database schema, sample of data from the tables, already written partial query structure and semantic information on the schema. These information can be effectively used to give auto suggestions to the user who is writing the query.

We present a simple recommender which makes the process of query writing easier. Illustrating with an example, consider the following query and associated recommendations: SELECT (suggest possible clauses to start the query) Orders.OrderID, Customers.CustomerName, Orders.OrderDate ( No recommendations or as-you-type suggestions; incase of multiple column names, give preference to column within the same table) FROM Orders (autofill table name based on suffix of SELECT Statement; incase of multiple table names, give preference to tables containing all the columns within

itself over tables with join possibilities) INNER JOIN Customers (suggest possible tables on which inner join could be done based on the data from Orders table) ON Orders.CustomerID=Customers.CustomerID (suggest possible join predicate based on tables and column names);

In this paper we discuss some observations which have helped us rank the suggestions, the implementation of the tool, and the tool‘s evaluation using the standard Northwind database. We present a schema-aware recommendation system for SQL queries with a simple instant-response user interface. To evaluate the tool, we retrace the steps of some common queries used on Northwind database and evaluate the usefulness of the query recommendations given by the system.

# 2. IMPLEMENTATION

Typically, the body of a paper is organized into a hierarchical structure, with numbered or unnumbered headings for sections, subsections, sub-subsections, and even smaller sections. The command `\section` that precedes this paragraph is part of such a hierarchy.[1] LaTeX handles the numbering and placement of these headings for you, when you use the appropriate heading commands around the titles of the headings. If you want a sub-subsection or smaller part to be unnumbered in your output, simply append an asterisk to the command name. Examples of both numbered and unnumbered headings will appear throughout the balance of this sample document.

Because the entire article is contained in the **document** environment, you can indicate the start of a new paragraph with a blank line in your input file; that is why this sentence forms a separate paragraph.

## 2.1 Global Index Data Structure

The implementation is irrespective of the type of database used underneath. A global index structure is populated which contains the schema information from database when a connect statement is detected. This is simply an object with sufficient methods and attributes to represent a schema and recommend on them. An overloaded âĂŸpopulateâĂŹ method is written for each of the supported types of databases(currently only Mysql is supported) to populate this index. Hence adding support for new databases becomes relatively straightforward. The schema information essentially contains the table names, column names and datatype and key information for each of the tables. Apart from these a fraction of data is also collected from these Each database has its own way of extracting schema through the populate method. But the index structure just contains the information that is common to all relational databases in the market. Hence it simply becomes a matter of a few lines of code into the well defined interface of DBindex populate method to extend the feature to any databases.

## 2.2 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

### 2.2.1 Inline (In-text) Equations

A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin. . .\end` construction or with the short form `$. . .$`. You can use any of the symbols and structures, from $\alpha$ to $\omega$, available in LaTeX[?]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\to\infty} x = 0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

### 2.2.2 Display Equations

A numbered display equation – one set off by vertical space from the text and centered horizontally – is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\to\infty} x = 0 \tag{1}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty} x_i = \int_0^{\pi+2} f \tag{2}$$

just to demonstrate LaTeX's able handling of numbering.

## 2.3 Citations

Citations to articles [?, ?, ?, ?], conference proceedings [?] or books [?, ?] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the proper location in the `.tex` file [?]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide*[?].

This article shows only the plainest form of the citation command, using `\cite`. This is what is stipulated in the SIGS style specifications. No other citation format is endorsed.

## 2.4 Tables

[1]This is the second footnote. It starts a series of three footnotes that add nothing informational, but just give an idea of how footnotes work and look. It is a wordy one, just so you see how a longish one plays out.

**Table 1: Frequency of Special Characters**

| Non-English or Math | Frequency | Comments |
|---|---|---|
| Ø | 1 in 1,000 | For Swedish names |
| $\pi$ | 1 in 5 | Common in math |
| \$ | 4 in 5 | Used in business |
| $\Psi_1^2$ | 1 in 40,000 | Unexplained usage |

**Figure 1: A sample black and white graphic (.eps format).**

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

## 2.5 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of **.eps** and **.ps** files to be displayable with LaTeX. More details on each of these is found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper "floating" placement of tables, use the environment **figure*** to enclose the figure and its caption.

Note that either **.ps** or **.eps** formats are used; use the `\epsfig` or `\psfig` commands as appropriate for the different file types.

## 2.6 Theorem-like Constructs

**Figure 2: A sample black and white graphic (.eps format) that has been resized with the `epsfig` command.**

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. There are two forms, one produced by the command `\newtheorem` and the other by the command `\newdef`; perhaps the clearest and easiest way to distinguish them is to compare the two in the output of this sample document:

This uses the **theorem** environment, created by the `\newtheorem` command:

THEOREM 1. *Let $f$ be continuous on $[a, b]$. If $G$ is an antiderivative for $f$ on $[a, b]$, then*

$$\int_a^b f(t)dt = G(b) - G(a).$$

The other uses the **definition** environment, created by the `\newdef` command:

*Definition 1.* If $z$ is irrational, then by $e^z$ we mean the unique number which has logarithm $z$:

$$\log e^z = z$$

Two lists of constructs that use one of these forms is given in the *Author's Guidelines*.

and don't forget to end the environment with figure*, not figure!

There is one other similar construct environment, which is already set up for you; i.e. you must *not* use a `\newdef` command to create it: the **proof** environment. Here is a example of its use:

PROOF. Suppose on the contrary there exists a real number $L$ such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c} \left[ gx \cdot \frac{f(x)}{g(x)} \right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

Complete rules about using these environments and using the two different creation commands are in the *Author's*

**Table 2: Some Typical Commands**

| Command | A Number | Comments |
|---|---|---|
| `\alignauthor` | 100 | Author alignment |
| `\numberofauthors` | 200 | Author enumeration |
| `\table` | 300 | For tables |
| `\table*` | 400 | For wider tables |

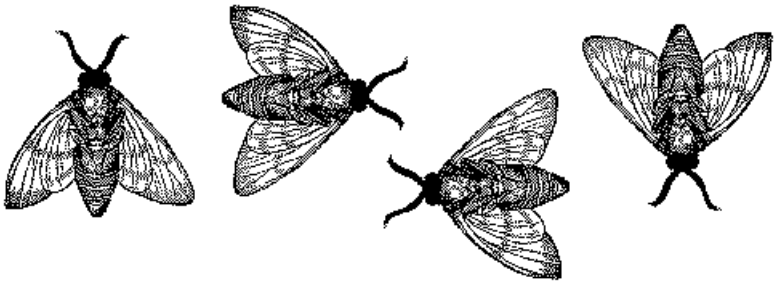

**Figure 3: A sample black and white graphic (.eps format) that needs to span two columns of text.**

*Guide*; please consult it for more detailed instructions. If you need to use another construct, not listed therein, which you want to have the same formatting as the Theorem or the Definition[**?**] shown above, use the `\newtheorem` or the `\newdef` command, respectively, to create it.

### A *Caveat* for the TEX Expert

Because you have just been given permission to use the `\newdef` command to create a new form, you might think you can use TEX's `\def` to create a new command: *Please refrain from doing this!* Remember that your LATEX source code is primarily intended to create camera-ready copy, but may be converted to other forms – e.g. HTML. If you inadvertently omit some or all of the `\def`s recompilation will be, to say the least, problematic.

## 3. CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LATEX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

## APPENDIX

## A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

### A.1 Introduction
### A.2 The Body of the Paper
#### *A.2.1 Type Changes and Special Characters*
#### *A.2.2 Math Equations*

*Inline (In-text) Equations*

*Display Equations*

#### *A.2.3 Citations*
#### *A.2.4 Tables*
#### *A.2.5 Figures*
#### *A.2.6 Theorem-like Constructs*
*A Caveat for the TEX Expert*
### A.3 Conclusions
### A.4 Acknowledgments
### A.5 Additional Authors

This section is inserted by LATEX; you do not insert it. You just add the names and information in the `\additionalauthors` command at the start of the document.

### A.6 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references) to create the .bbl file. Insert that .bbl file into the .tex source file and comment out the command `\thebibliography`.

## B. MORE HELP FOR THE HARDY

The acm_proc_article-sp document class file itself is chockfull of succinct and helpful comments. If you consider yourself a moderately experienced to expert user of LATEX, you may find reading it useful but please remember not to change it.