

# NakeDB: Database Schema Visualization

Luis Miguel Cortés-Peña, Yi Han, Neil Pradhan, Romain Rigaux

**Abstract**—Current database schema visualization tools are ineffective and hard to use. Since the documentation of a database is often out of date or static, a dynamic visual tool that can generate real-time database schema visualization would greatly assist software engineers working on these databases. This paper illustrates different methods and tools for database schema visualization. The paper then proceeds to present the preliminary findings of our database visualization tool, namely NakeDB. This tool is a Java tool built using the Prefuse toolkit which enables the visualization of potentially huge databases (e.g. >1000 tables). In order to read the structure of databases, the schema is parsed into XML files that can be processed by Prefuse. The major features of NakeDB includes searching for a particular node or set of nodes; dynamic filtering of node depth to “restrict” the mass of information; and visualization techniques such as overview and details, zooming and panning, and focus and context. NakeDB offers four different layout views(force directed, circular, radical tree, node-link tree) and four different distortion techniques(no distortion, free to move, fisheye, bifocal). Finally, our tool allows the encoding of different table attributes into node color, shape, size of node, type of label and visual aggregation which can be set to represent table size, table type, or criticalness. NakeDB was evaluated by four different users and proved to be a useful tool for visualizing complex database relations.

**Index Terms**—NakeDb; InfoVis; Database; Visualization; Prefuse; Filtering; Data Encoding; CS 7450; Georgia Tech;

## I. INTRODUCTION

**D**ATABASE systems consists of a collection of information organized so that it can easily be accessed, managed and updated. Such systems are commonly used in many areas such as inventory management, financial services and web services. These database systems handle large amounts of data. Data contained in the database is stored in the form of tables. As the database grows, the number of such tables increase. The increase in the data also results in complex relationships between tables. As the database’s number of tables increase, it becomes time consuming to understand the flow between these tables. This is especially true if no documentation is available. We have created NakeDB - a database schema visualization tool created using the Prefuse toolkit [1], [2]- which aims at providing insight in comprehending complex database relationships.

This paper is structured as follows. A background is given on the current problems of database visualization. NakeDB is then introduced. The different layouts implemented are shown, and their advantages and disadvantages are given. We then show the different methods of interacting with large amounts of data. Visual encodings used in NakeDB are then explained. We then provide different case scenarios explaining how our tool is useful. Our evaluation procedure and results is then provided. Finally, we conclude and mention some possible enhancements that can be made to our software.

## II. BACKGROUND

Information visualization techniques are used to present data to users in the form of images. Representing data in a pictorial format helps in its interpretation as this method occurs fast because of the parallel processing capabilities of the eyes. A good selection of visual representation format can display large amount of information. In addition, it is also possible to interact with the visualization and “drill down” into the dataset to focus on a region of interest. Such techniques are of interest to database users as they can make databases easier to use. Various representations are used for laying out different relationships. Such representations include: Entity Relationship diagram, Use Case diagrams and DB ER diagrams. An example of a DB ER diagram is provided in Figure 1. These representations, however, do not provide a comprehensive representation of the database schema. In addition, since these diagrams lack the ability to allow interaction, they do not facilitate exploration of data. An interactive visual tool that can generate real-time database schema visualization would greatly assist software engineers in understanding a database.

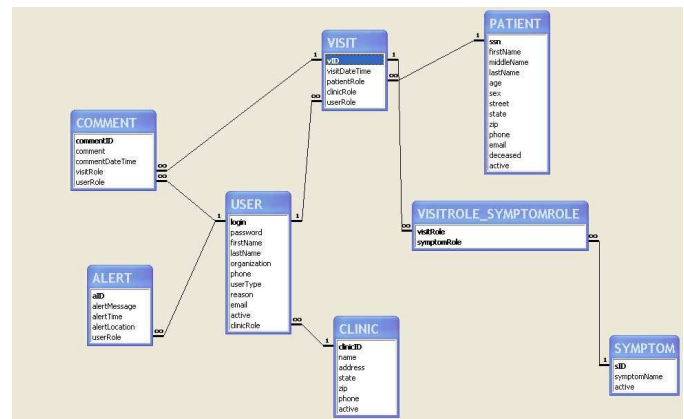


Fig. 1. Example of a DB ER diagram.

As a result, we have built a dynamic database visualization that provides users with different views of the tables to help them understand the complex relationships between them. The visualization can aid a user in locating bad designs such as possible loops in table relations. By using tool tips on tables, statistics such as number of connections to and from a table can be displayed. Also implemented is visual encoding to illustrate groups of table, frequency of access, amount of data and number of connections. The user can interact with the visualization to search for a table from amongst thousands of tables. He/she can also perform zooming and panning in order to focus into the table information of the database.



various layouts to help the user comprehend the data. The different layouts implemented include: node-link tree layout, circular layout, force directed layout, and radical tree layout. All these layouts have their own advantage and disadvantages [6]. The implementations of these are further discussed in the following sections.

1) *NodeLink Tree Layout*: The node-link tree layout for the database schema is as shown on Figure 6. As can be observed, each node represents a table in the database. Tables are related by foreign keys, which store an ID number from another table. This node-link tree layout provides a systematic way of visualizing these relationships. Some advantages and disadvantages include:

- *Advantage* - Clear view of all the tables and their relations.
- *Disadvantage* - Not scalable.

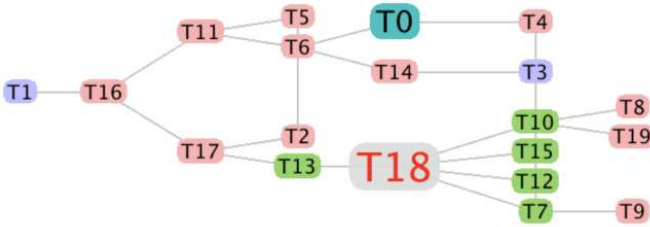


Fig. 6. NakeDB showing Node-link tree layout.

2) *Circular Layout*: In this layout the nodes are arranged along the periphery of a circle as shown in Figure 7. Some advantages and disadvantages include:

- *Advantage* - Clear circular layout of the nodes.
- *Disadvantage* - The edges overlap which makes it difficult to comprehend the relationships.

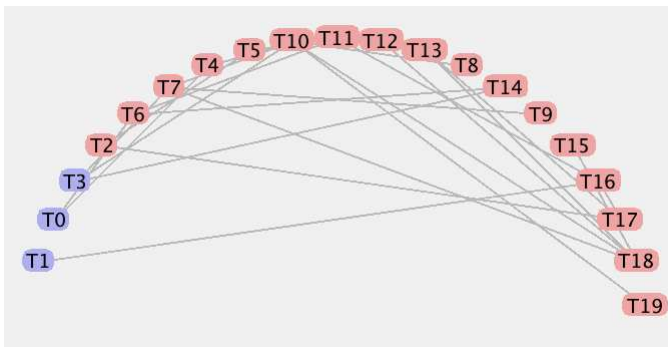


Fig. 7. Circular layout as seen in NakeDB.

3) *Force Directed Layout*: In the force directed layout, nodes are forced to arrange themselves according to the group they belong to (see Figure 8). Some advantages and disadvantages of this type of layout include:

- *Advantage* - Uses the space efficiently for a clear layout.
- *Disadvantage* - As the number of nodes increase, nodes tend cluster together.

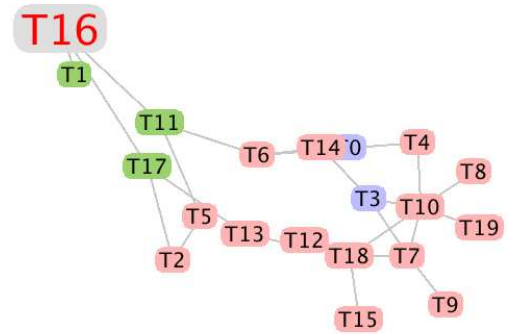


Fig. 8. Force directed layout.

4) *Radical Tree Layout*: Radical view shows nodes spaced out as shown in Figure 9. Advantages and disadvantages include:

- *Advantage* - Clear view of all the nodes and relations.
- *Disadvantage* - Bad presentation of structure.

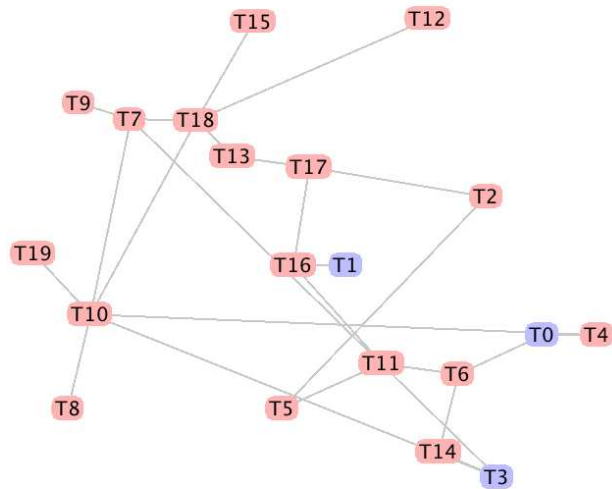


Fig. 9. Radical Tree Layout.

## B. Interaction Techniques

We have implemented various interaction techniques which can aid the user interact with the data and the visualization. These include search boxes, drop down menus, and arrow scrolls to generate dynamic queries. Such techniques help in answering “what - if” type of questions that help a user find insights. Insights help users define new questions, hypothesis and models for their data [7]. Figure 10 shows a screen capture of such interaction techniques.

1) *Zooming and Panning*: In the case that the visualization contains large number of nodes, the user can use the zooming effect to converge to a particular part of the visualization.

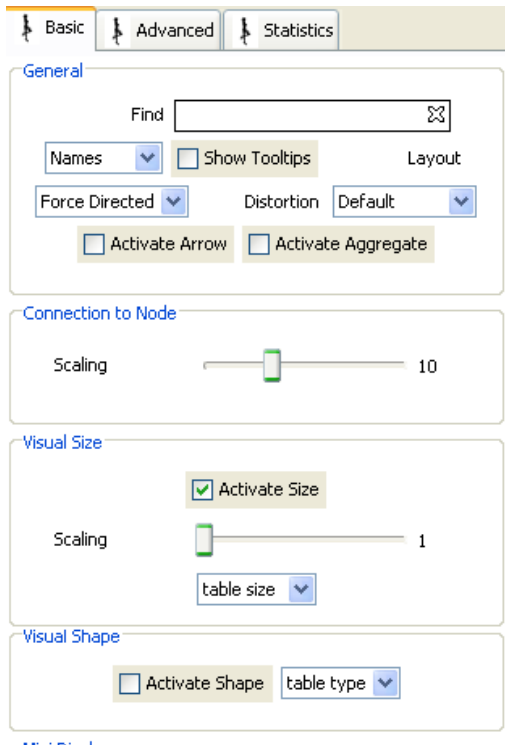


Fig. 10. Screenshot of interaction techniques implemented in NakeDB.

Zooming is done through the scroll function on the user’s mouse. Panning can be used to move to other parts of the visualization. The mini display can be used to understand the current position in the visualization.

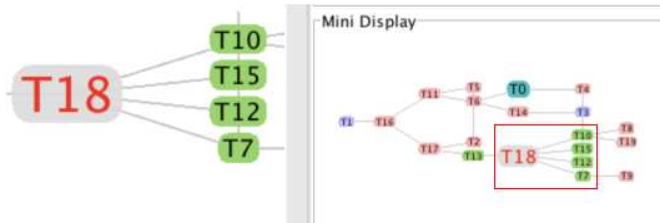


Fig. 11. Screen shot showing NakeDB’s Mini Display.

2) *Focus and Context*: Another implementation done for handling the case of large number of nodes is the use of focus and context. This allows a user to understand the context of the magnified nodes with respect to other nodes in the visualization. The tool uses the following distortions for focus and context:

a) *Fisheye*: As shown in Figure 12, the center of the field of view is magnified with a continuous fall off in magnification towards the edges.

b) *Bifocal Distortion*: Figure 13 shows a screen capture of the bifocal implementation in NakeDB. Bifocal distortion consists of spatially separating the focused item from the remaining items.

3) *Search Filter*: NakeDB has search filter capabilities implemented. This is particularly useful when a software programmer knows that a particular table exists but cannot

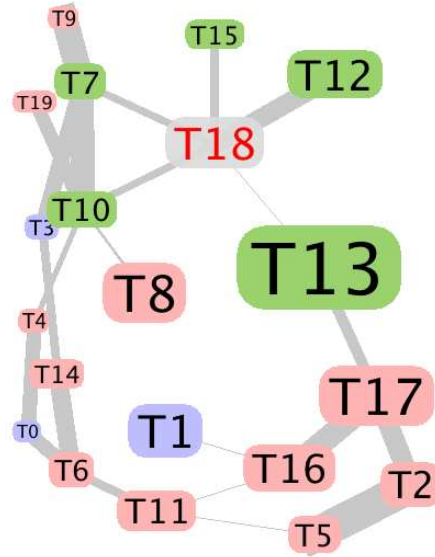


Fig. 12. Fisheye View.

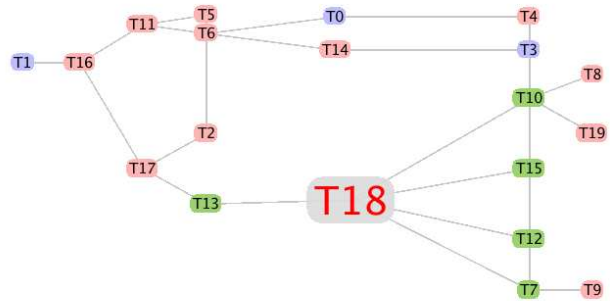


Fig. 13. Bifocal Distortion

remember the table’s full name or is not aware of the connections going into or out of the table. In such a scenario a user can use the search filter within the visualization as shown in Figure 14. The user gives a search in the search box (i.e. for "T0"). The search results show as "1 match" found and that particular node is highlighted.



Fig. 14. Search functionality implemented in NakeDB.

C. *Tooltip*

The visualization tool has a tooltip feature to show information about a particular table. The user takes the cursor over a node and a small rectangular box appears which contains information about the node like size of the table and the

attributes of the table (see Figure 15). This tool is useful as users can obtain detailed information at hand.

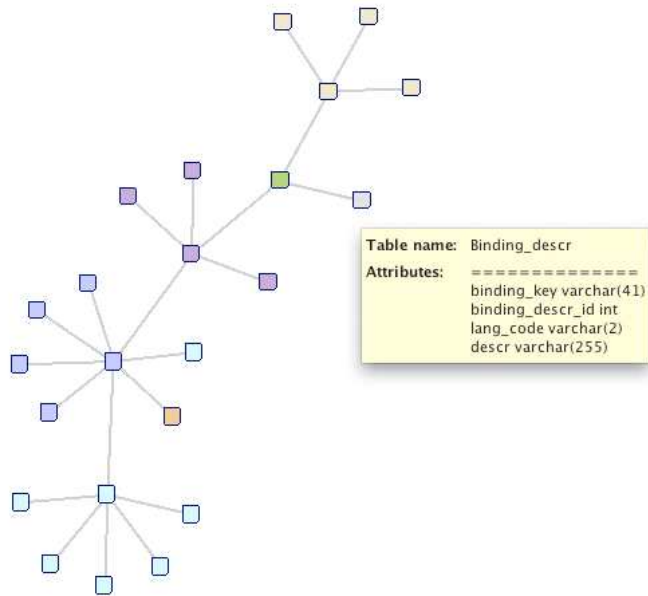


Fig. 15. NakeDB showing tooltip functionality.

D. Full Text Labels

In addition to the tooltip feature, NakeDB also has the capability of displaying this information in as the node's label. This allows the user to view a more detailed view of the attributes of all tables in the database. An example is shown in Figure 16.

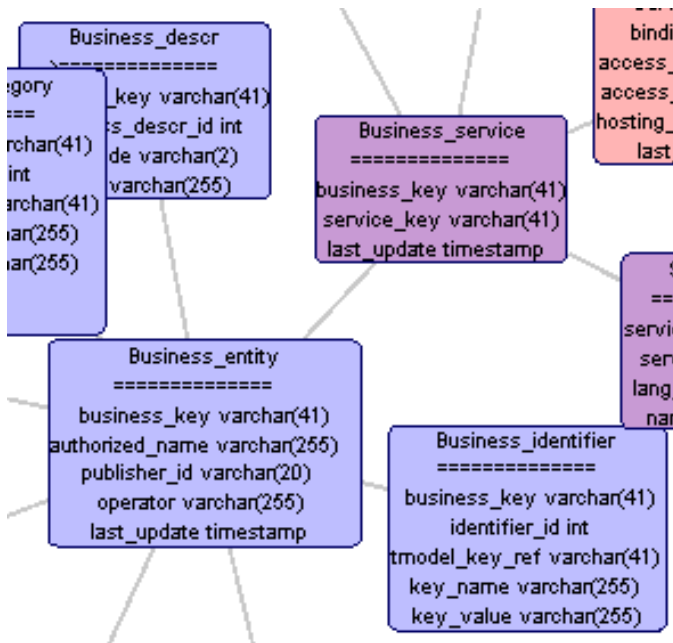


Fig. 16. NakeDB is capable of fully displaying attributes in the node's label.

E. Distance Filter

This feature allows the user to visualize how far a particular node is away from another node. The user selects a particular node and then changes the scaling factor in the interaction panel on the right. For example, Figure 17 shows nodes with a distance filter of four (left) and a distance filter of two (right).

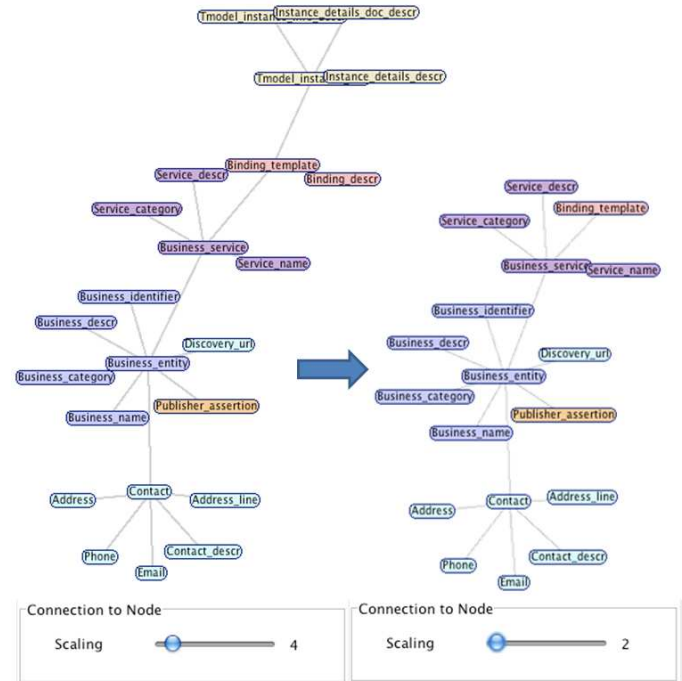


Fig. 17. NakeDB after changing the distance filter from four (left) to two (right).

F. Table Size

The user can change the scaling of the table size located in the interaction panel (see Figure 18). This option allows the user to change the node size in the visualization tool. The greater the scaling factor is, the greater size of the node is. This helps users in separating nodes of particular data size.

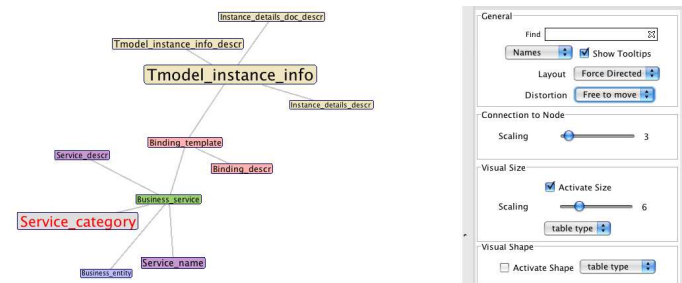


Fig. 18. Screen capture of NakeDB showing Table Size functionality.

G. Visual Encoding

Different visual encoding techniques were implemented into NakeDB in an attempt to give more information about the database schema. Our tool allows data to be encoded as:

- Color
- Size
- Shape
- Stroke
- Position
- Visual Aggregation

As shown in Figure 19, a node can be represented in a different shape such as a circle, diamond, star, plus sign and rectangle. The tables which are related are shown to have the same shape. This allows grouping of tables. Another practical encoding technique is to use visual aggregation. Using this technique, the user himself can define groupings for different nodes by dragging these nodes into predefined areas. Nodes in one area would then represent a group. This is particularly useful for frequently changing databases. An example of such aggregation is shown in Figure 20. The screen shot displayed in Figure 20 also shows the use of arrows for directionality. An example of how the data can be encoded is given in Table I.

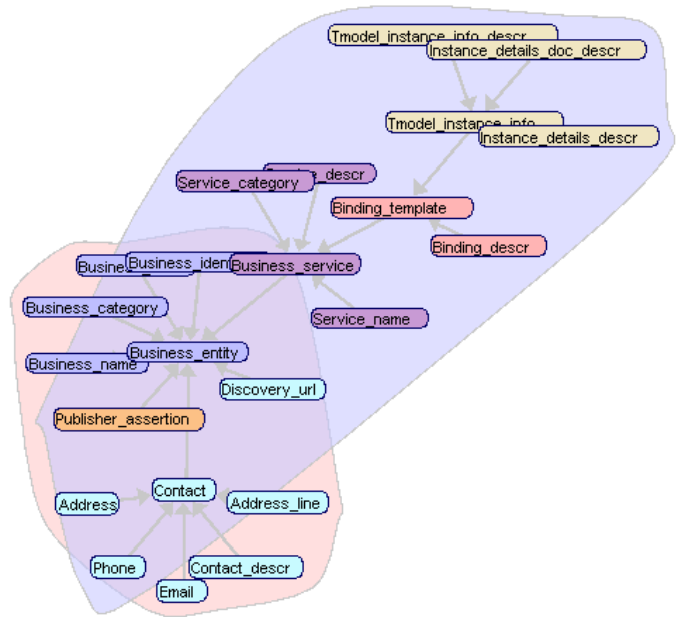


Fig. 20. Screen capture of NakeDB showing visual aggregation.

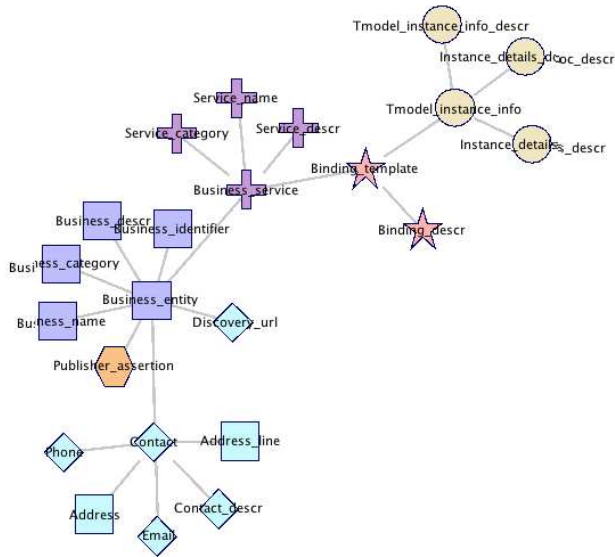


Fig. 19. Screen capture of NakeDB showing encoding with shapes.

TABLE I  
EXAMPLE OF ENCODING INFORMATION

| Encoding           | Representation                       |
|--------------------|--------------------------------------|
| Size               | Amount of data in the table          |
| Color              | Group similar tables                 |
| Shape              | Group tables with similar attributes |
| Visual Aggregation | User defined groups                  |

*H. Neighbor Highlighting*

If the visualization consists of thousands of nodes then determining the relationships between different nodes becomes difficult. The most common task in such a scenario is finding the neighbors of a node. In NakeDB this can be achieved by

selecting the node and enabling the control of highlighting the neighbors of the node. The results is as shown in Figure 21.

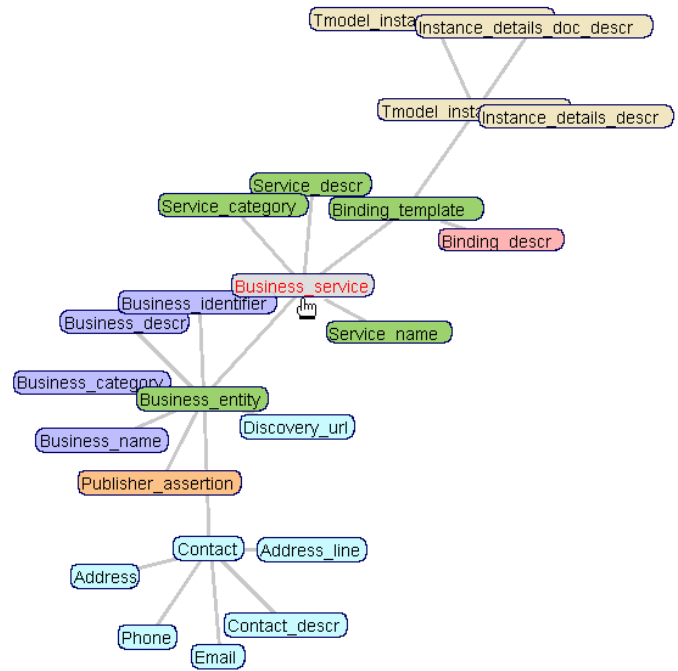


Fig. 21. Screen capture of NakeDB showing Neighbor Highlighting.

*I. Additional Features*

Other features of NakeDB include those shown in Table II.

V. NAKEDB: CASE SCENARIOS

The visualization tool was designed to be useful for software programmers and database designers for a number of

scenarios. The following are a subset of examples in which NakedB would be useful.

#### A. Scenario 1

Consider a scenario where a user working for a company and the user's project involves a database with large amount of tables. In order to effectively and efficiently use the database, an overview of the different parts of the database is needed. By only looking at the database schema, it is difficult to see the connection amongst the tables and if a new table or a foreign key is added, it could seem incoherent. In this visualization the partitions and shards are represented with a color coding and group filters. It enables to select the part of the database the user wants and then analyze the relationships and attributes of the tables. With our tool, it is easy to reduce the number of tables to only the few that are useful.

#### B. Scenario 2

Imagine a user joining a team working on a Web 2.0 website and need to take care of the backend of the website. In such a scenario, watching the SQL of the table does not provide an obvious overview of the schema. In a few seconds, the schema can be converted to a XML file that can be loaded into NakedB. All the tables and structure appear on the screen. It provides the user with the "big picture" of the database. The user can now zoom in/out, pan, filter tables, and navigate freely within the schema to get a helpful visual representation of the database.

#### C. Scenario 3

Consider a scenario where a user is in charge of optimizing a company's website. In such a case, many improvements can be made to the database. NakedB provides a fast visualization of the whole schema. Potential problems and list of connections can be determined. New paths and bottlenecks can be visually spotted thanks to the tool.

#### D. Scenario 4

A user is in a team managing a big application with a large database. The tool enables to showoff the complexity of a database since it provides a broad view of the importance of the size of the data. In order to expose the importance of the work, a demo can be shown to display different parts of the database, zoom in / out and show it to the exterior.

TABLE II  
ADDITIONAL FEATURES IMPLEMENTED IN NAKEDB

| Features                | Description  |
|-------------------------|--|
| Statistics Tab          | Shows different statistics for the table such as size and attributes of table. |
| Zoom                    | The user can zoom into the visualization using the mouse wheel.                |
| Zoom to fit             | This feature allows the user to fit the visualization in a single screen.      |
| Enable / Disable arrows | This feature allows the user to see directions in the visualization.           |

## VI. EVALUATION

Four persons were chosen at random to evaluate NakedB. These are referred to as subject A, subject B, subject C, and subject D. They were offered coke and pizza as compensation. The steps taken for evaluation are as follows:

- 1) Fill the database background knowledge form.
- 2) Show quick overview of NakedB.
- 3) Let users utilize the tool for a few minutes.
- 4) Ask users to accomplish five tasks.
- 5) Fill the questionnaire.

#### A. Database Knowledge Form

In order to place users and results in context, it was important to know what level of database knowledge did the users have. Questions asked in the database background knowledge form are:

- 1) How often do you work with databases? (1) Not at all, (2) Not really, (3) Some what, 4 (Often), 5 (Very much)
- 2) What is the typical size of database that you work with? (0) None, (1) 10 tables, (2) 100 tables, (3) 500 tables, (4) >500 tables
- 3) What are the ways that you use to understand relationships between these tables? (1) DB schema diagram, (2) DB schema files, (3) DBMS tools, (4) Other
- 4) So tell me more about hte worst experience you have had handling such huge databases?
- 5) What are the ways in which you solve such problems?
- 6) Have you ever used any visualization tool? If so, which one?

#### B. Tasks

In order to evaluate our tool, a series of simple tasks were created so that we could see how our tool compared with user's goals. The tasks asked to be performed by the users are as follows:

- 1) Can you locate table T19?
- 2) Name the table with maximum data.
- 3) do you think table T15 is related to Table 19?
- 4) What are the attributes of table T10? (Hint: You may want to change the view)
- 5) Find the table with size 4000. How far is it (w.r.t. levels) from Table T5?
- 6) Browse through the visualization tool and write in brief about the most interesting aspect of the tool.

#### C. Questionnaire

After the users performed the evaluation tasks, a series of concluding questions were ask in order to get user feedback. The questionnaire included the following set of questions:

- 1) How many tasks did you complete in 20 seconds?
- 2) Which task do you think you would probably use the most?
- 3) Which layout did you find most useful? why? (1) Node - link, (2) Circular, (3) Force Directed, (4) Radical
- 4) Which distortion did you find the most useful? why? (1) Fisheye, (2) Bifocal, (3) Free to move, (4) No distortion

- 5) What is your opinion about the panel for interaction techniques? (1) Very Easy, (2) Easy, (3) Not Hard nor Easy, (4) Hard, (5) Very Hard
- 6) Did you use the scales for data size and connection level present in the interaction panel? If so where did you find them to be most useful?
- 7) In general, do you think such a tool would be useful?
- 8) Could you mention some additional tasks that could be useful with such a tool?
- 9) If you were given an opportunity to change something in this tool, what would it be?
- 10) Is there any information which you would like to add which you think you might not have said?

#### D. Evaluation Results and Analysis

Table III and Table IV shows some results of the database knowledge form and the questionnaire respectively. All subjects, with the exception of subject D, thought the visualization to be useful. Subject D, however, had never worked with a database before. Most participants were able to complete most of the tasks under 20 seconds. It was found that node-link and force directed layouts are the most useful compared to radical and circular layouts. Three out of four participants thought the zoom distortions (fisheye, bifocal) were ineffective since these distortions creates jitters as the mouse moves. It was also found that the table size scaler, which allows users to control the size encoding, was useful by all participants. Overall, NakedB showed to be a useful tool for the visualization of database schema.

TABLE III  
DATABASE KNOWLEDGE QUESTIONS AND ANSWERS

| Question | A          | B          | C          | D          |
|----------|------------|------------|------------|------------|
| 1        | Some what  | Some what  | Very much  | Not at all |
| 2        | 100 tables | 100 tables | 100 tables | None       |
| 3        | 1          | 2,3        | 1,3        | N/A        |

TABLE IV  
POST-INTERVIEW QUESTIONS AND ANSWERS

| Q | A               | B                  | C                 | D              |
|---|-----------------|--------------------|-------------------|----------------|
| 1 | 5               | 5                  | 5                 | 4              |
| 2 | 4,5             | 1,3,4              | 2                 | 2              |
| 3 | Force directed  | Node-link          | Node-link         | Force directed |
| 4 | Free to move    | No distortion      | Bifocal           | Free to move   |
| 5 | Easy            | Hard               | Not hard nor easy | Very hard      |
| 6 | finding extrema | finding sizes      | finding sizes     | Yes            |
| 7 | Yes             | Yes                | Yes               | No             |
| 8 | NA              | Simulating queries | Checking errors   | NA             |

## VII. RELATED WORK

As mentioned earlier, people from the database community realize the importance of database visualization and hence several works are going on in this field. One such work includes SchemaSpy which is a Java based tool that analyzes the metadata of the schema in the database and generates visualizations [8]. Figure 22 shows an example output of SchemaSpy. As shown, SchemaSpy allows browsing through the hierarchy of tables consisting of child and parent.

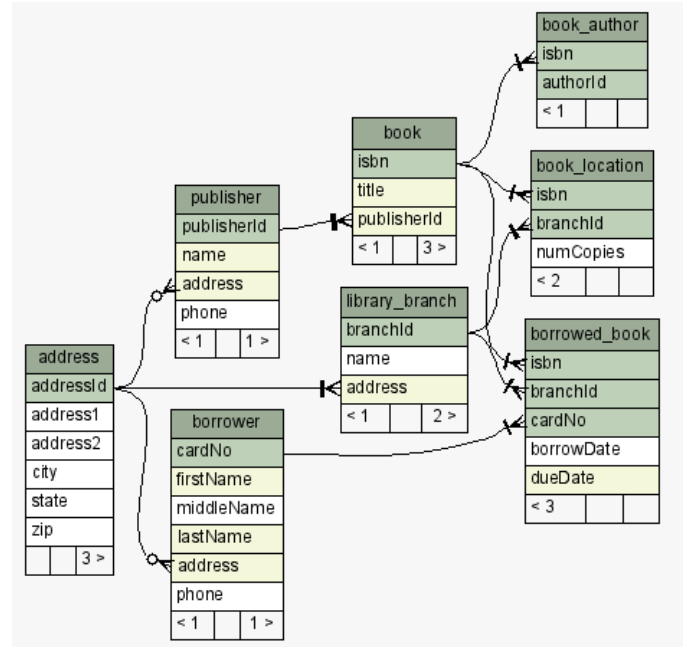


Fig. 22. Example output of SchemaSpy.

Another database visualization tool is Schemaball [9], shown in Figure 23. Its main objective is to display relationships between tables. It arranges the tables along a circle and shows table relationships using straight or curved lines.

These visualizations, however, lack of the flexibility provided by NakedB which allows for users to interact with data in different ways.

## VIII. CONCLUSION AND FUTURE WORK

While many representations have been developed for visualization of database schema, most of them generate a static layout. This does not allow exploration of data to find patterns in them. NakedB allows dynamic interaction of database schema. Our tool is capable of encoding using node color, size, shape, stroke, and visual aggregation. NakedB also allows for visualization using different layouts such as node-link tree layout, circular layout, force directed layout, and radical tree layout. It can also display high number of nodes using overview and details, zooming and panning, and focus and context. The search and dynamic filtering capabilities are other powerful features of NakedB. We have given examples of how this tool can be useful for users working with databases. Our tool was evaluated with four different users and it can be concluded that NakedB is a useful tool to gain insights on the database schema.

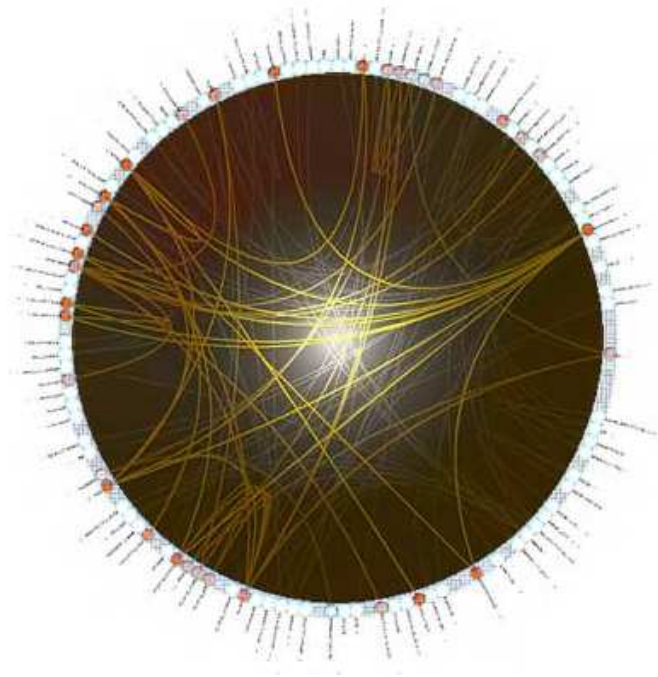


Fig. 23. Schemaball showing a visualization of a database.

The application can be further developed to have a visual interface to generate SQL queries on the tables displayed and display such results. Another useful feature mentioned by one of the participants that evaluated NakedB is to be able to view history of queries and other database logs. Another participant mentioned fading nodes not relevant to a search result.

#### IX. ACKNOWLEDGMENTS

The NakedB visualization tool was developed as part of the Information Visualization (CS 7450) course offered at Georgia Tech on Spring 2008. We would like to thank Professor Jim Foley and teaching assistant Steven Paul for their constant support and valuable feedback. We would also like to thank Jeff Heer the author of the Prefuse toolkit as this was extensively used in the development of our work.

#### REFERENCES

- [1] J. Heer, S. Card, and J. Landay, "prefuse: a toolkit for interactive information visualization," *Conference on Human Factors in Computing Systems*, pp. 421–430, 2005.
- [2] "prefuse — interactive information visualization toolkit," <http://prefuse.org/>, April 2008.
- [3] "Communicate - prefuse assistance pool PAP," <http://goosebumps4all.net/34all/bb/forumdisplay.php?fid=18>, April 2008.
- [4] "SourceForge.net: Forums for the prefuse visualization toolkit," [http://sourceforge.net/forum/?group\\_id=98962](http://sourceforge.net/forum/?group_id=98962), April 2008.
- [5] B. Shneiderman, "Dynamic queries for visual information seeking," *Software, IEEE*, vol. 11, no. 6, pp. 70–77, 1994.
- [6] I. Herman, G. Melançon, and M. Marshall, "Graph Visualization and Navigation in Information Visualization: A Survey," 2000.
- [7] J. van Wijk and T. Eindhoven, "The Value of Visualization," *Visualization, IEEE 2005*, pp. 11–11, 2005.
- [8] "SchemaSpy," <http://schemaspy.sourceforge.net/>, April 2008.
- [9] "Schemaball," <http://mkweb.bcgsc.ca/schemaball/>, April 2008.