

# MAKING DATABASE SYSTEMS USABLE & WHY NOT?

Nikhil Prakash  
Rozemary Scarlat

# Making Database Systems Usable

by H.V. Jagadish, et al.

## Overview

- Methods to improve usability of a database
- Enable easier access to information with more accurate results
- Allow for non - technical users to effectively search through data
- Ensure results that are helpful

# Usability Characteristics

- Allow users to enter more sophisticated queries
- Return more precise and complete answers
  - All and only relevant results
- Users have expectations for results
- Databases need to be created and updated

# Human Error

## Innovative Query Interface

- Visual Interface
    - XQBE, MIX, XING, VISIONARY, etc.
  - Text Interface
    - the use of keyword search
- ## Database Personalization
- Structure the database based on individual concerns

# MiMI

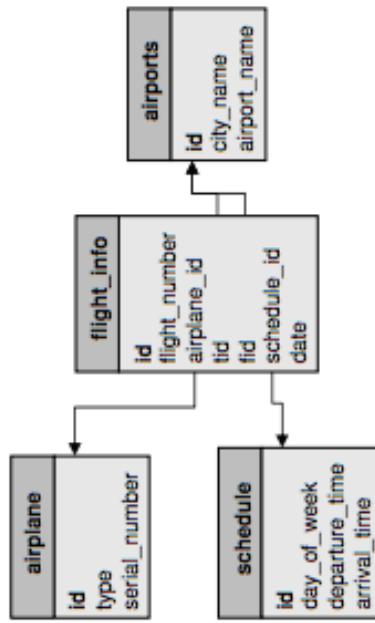
- An integration of protein interaction databases
  - Based on the human genome project
- Most of the intended users were people without any database background
- Two query interfaces: XQuery and Mquery
  - MQuery - queries can be made by selecting points on graphs
- Syntax of language was not an issue, but problems with formulating queries

# Accomplishments/Problems

- Development of a number of interfaces to access MiMI
  - Autocompletion interface
  - An interactive Natural Language Interface for Querying XML
- NaLiX - an interface that is not just used for selection, but also joins, nests, and aggregates
- Major Problems
  - Inconsistencies between different interfaces
  - No feature to explore and manipulate the data
  - Data cannot easily be updated

# Normalized Relationship Representation

- ❑ Organizing the database by linking the information
  - ❑ Reduces redundancy and dependency
- ❑ Example: Airline database
  - ❑ Join flights to airports



# Query Formulation Options

- Too many choices is a bad thing
  - "regret for path not taken", "need for closure", "exercising options is not costless", etc.
- In SQL, many times the same query can be made in multiple ways allowing the user multiple options
- Relational algebra only allows for a single option
  - Formulating the query is more difficult

# Unexpectedness

- Unable to Query
  - System fails to allow the user to correctly enter a query
  - Rigidity in query format or requirement
  - Forms - based interfaces
- Unexpected Results
  - Important to understand the reason for incorrect result
  - Can be an issue if user does not know what type of result to expect

# WSIWYG

- Structure of the query search hidden from the user
- The user often does not know how the query is being interpreted, which can create issues with obtaining the correct information
- WSIWYG allows for users to better understand how the database works
- Instantaneous- response interface lets the user to understand the search

# Database Creation and Updates

- Database structure should be more dynamic
- Users initially do not know what the final design will be, so the structure should not be rigid
- Example: Jane's shopping list transformation
  - Initially, a list of items and quantities, but later other attributes become important like price and occasion
- Information structures may have multiple types of information

# Presentation Data Models

- Geographic - group information by location
- Network - graph that is natural in many cases
- Multidimensional - different measure attributes represented in a multi-dimensional space
- Tabular - data simply represented in basic tables

# Discussion Points

- What are the trade-offs of balancing more options with accuracy? Pros and cons of each interface? Which interface seems to be most applicable?
- Would increasing transparency of the database allow for better results?
- What are the consequences of having a more dynamically structured databases?

# Why not? - Problem definition

- Why Not? is a series of statements about the potential reasons the data of interest to the user is missing from the result set

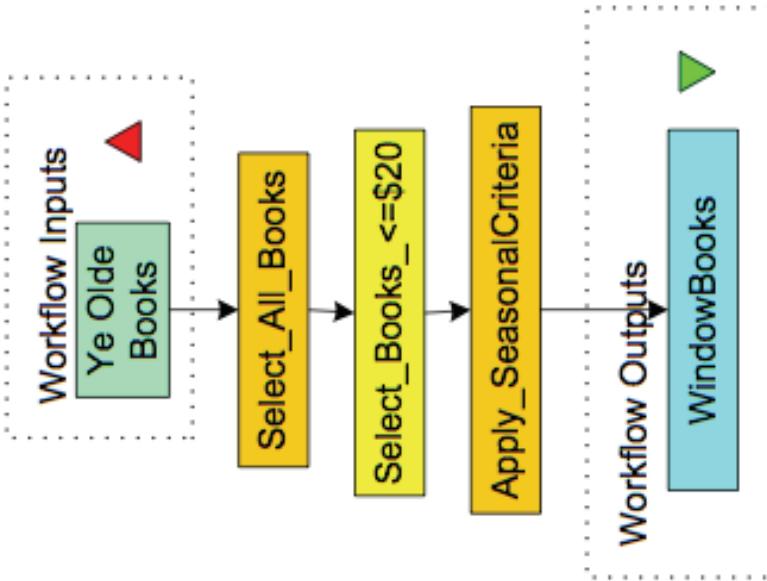
Author	Title	Price	Publisher
Euripides	Epic of Gilgamesh	\$150	Hesperus
	Medea	\$16	Free Press
Homer	Iliad	\$18	Penguin
Homer	Odyssey	\$49	Vintage
Hrotsvit	Basilius	\$20	Harper
Longfellow	Wreck of the Hesperus	\$89	Penguin
Shakespeare	Coriolanus	\$70	Penguin
Sophocles	Antigone	\$48	Free Press
Virgil	Aeneid	\$92	Vintage

Table 1: The set of books in *Ye Olde Booke Shoppe*.  
The different concepts and approaches are explained in the paper using a book dataset and a series of query examples

# Why not? - Example

- Example:

A user queries the database of a book store to list all window-books which are around \$20 and the result is (Euripides, "Medea"). If the user is interested in the book (Hrotsvit, "Basilius") and wants to know why this is not part of the answer, the progress of the (Hrotsvit, "Basilius") record is traced through all the manipulations performed on the (Euripides, "Medea") and the manipulations where the two records behave differently are considered a possible reason.



# Why not? - Foundations

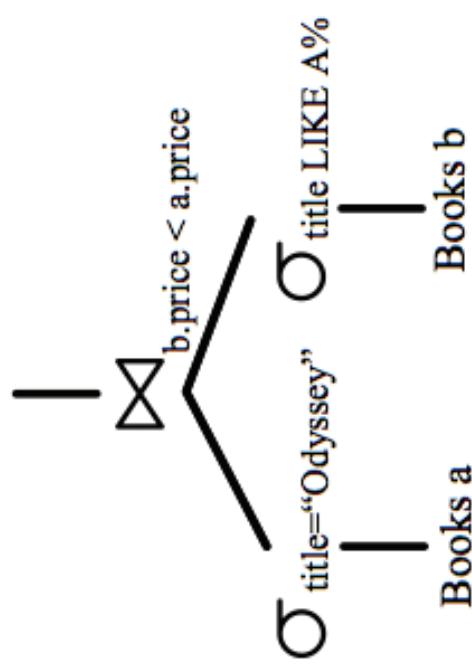
- Terminology:
  - **Data item** = the basic logical data unit, containing a set of attributes
  - **Dataset** = a set of data items
  - **MANIPULATION** = basic unit of processing in a workflow or query evaluation plan, that takes one or more datasets as input and outputs one dataset
- A query Q takes as input dataset D and gives a result set R. The user can ask then “why R does not contain any results satisfying predicate S”, which is expressed using positive predicate logic over a (sub)set of attributes of D.
- The **lineage** of data item  $d$  is the set of input data items that have influenced the inclusion of  $d$  in R.

# Why not? - Definitions

- A **data item is unpicked** if it that item has an attribute that appears in the predicate S and it is both not in the lineage of any item in R and is satisfaction-compatible with S.
- A data item  $d'$  is a **successor** of data item  $d$  if  $d'$  is in the output of a manipulation applied on an input dataset containing  $d$ .

# Why not? - Definitions (cont'd)

- A **picky manipulation** produces the exclusion the unpicked item in the manipulation's output set, even though the item (or one of its successors) was in the input set. All picky manipulations represent the complete answer to the Why not? question.
- A **frontier picky manipulation** is picky for at least one item in a set of unpicked items and in the same time there is no other item in that set which will later appear in the workflow.



# Why not? – Obtaining the answers

- The answer to the user's question "Why not items U?" is the set of manipulations which are frontier picky manipulations for the items in U.
- Two approaches:
  - Bottom-Up – checks the output of each manipulation from the beginning and when it finds unpicked successors marks it as picky manipulation; to identify the FPM it looks forward to make sure there are no more unpicked successors
  - Top-Down – starts with the outputs of the next to the last manipulation and checks the lineage of each data item to find successors to the unpicked items marking the last manipulation as the FPM; if no successors found it will go in reverse order to the next manipulation

# Why not? - Optimization

- Successor visibility - identifying the successor items can in many cases be done without storing the lineage or rerunning the manipulation
- One simple method to achieve successor visibility is attribute preservation
- As a general rule, projections, selections, joins and functions like MIN, MAX, COUNT, SUM, AVERAGE are successor visible

# Why not? - User satisfaction evaluation

- Considered as baseline an approach from related work, that tackle the Why not? question by defining trusted and untrusted sources and checking if there are any possible valid updates to the untrusted sources such that the data will be in the results.
- The main difference between the two approaches is that the "Non-answers" one tries to identify the attribute that caused the exclusion of a data item from the result set and this is the answer to the "Why not?" question
- They use a small user group (14 people), all with CS background ☺

# Why not? - Performance evaluation

- Used the Trio system which supports lineage tracing and evaluated with some specific crime queries
- Not clearly stated in the paper
- Top-Down approach seems to perform very good most of the times, and only slightly worse than bottom-up sometimes
- Successor Visibility optimization is great in most cases, but it performs extremely poor when it deals with a large number of tuples being passed through the manipulation

# Why not? - Discussion

- Useful in exploration of datasets by journalists with little CS knowledge
- In our projects:
  - Lineage
  - Query formulation
  - What else?