REDBUL: An Online System for Reverse Engineering of Relational Databases

D. Brdjanin, D. Vukovic, G. Banjac,

A. Kelec, I. Dujlovic, N. Obradovic, and D. Banjac

M-lab Research Group @ Faculty of Electrical Engineering

University of Banja Luka, Bosnia & Herzegovina

Presentation outline

- Motivation
- Research context and Related work
- UML-based approach to RDBS representation
- REDBUL implementation
- Reverse engineering example and tool demonstration
- Conclusion and future work

Motivation

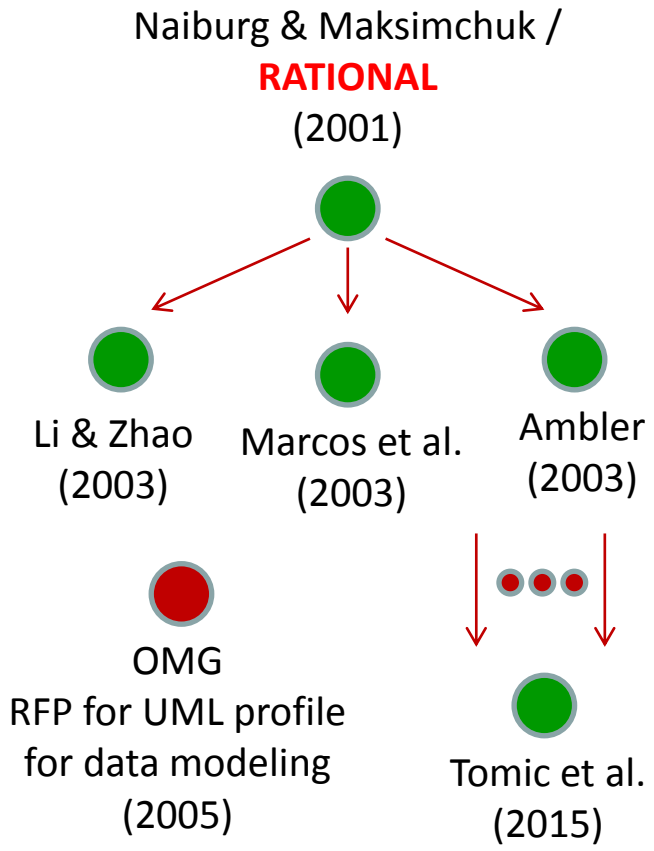
- The **relational model** is a leading model of database organization.
- The **overall structure and constraints** in the database constitute its **schema** (RDBS) – stored in a **data dictionary** within the database and managed by particular database management system (DBMS).
- **Reverse database engineering** involves **extracting and representing the corresponding RDBS** from an existing database.
- There **is no standardized approach** to represent the RDBS.
- UML class diagram is increasingly being, but **the existing approaches use specialized notation** (profiles).
- In this paper **we present an approach to reverse RDBS engineering and its visualization by the standard UML class diagram**.
- The **approach is implemented by REDBUL** – the **first online web-based tool for reverse RDBS engineering that uses the standard UML notation**.

Related work

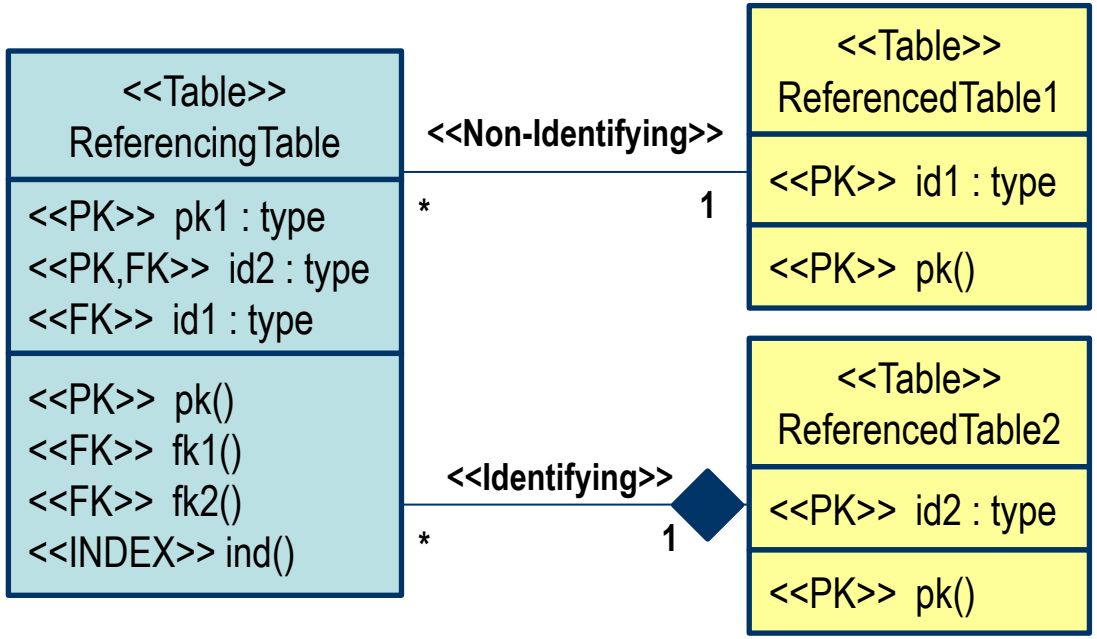
- Reverse database engineering has been the subject of research for many years – a number of different approaches & techniques aimed at extracting and visualizing the RDBS (IFO2, OMT, E-R, etc.).
- The most recent approaches are meta-model based – UML-based approaches are dominant.
- There is still no unified and standardized UML-based approach – the existing approaches mainly use specialized notation (i.e. profiles).
- In this paper we present **an approach to the RDBS representation by the standard UML class diagram** and the **corresponding online web-based tool for reverse RDBS engineering called REDBUL**.
- **REDBUL is the first online tool for reverse RDBS engineering, which uses the standard UML notation**.
- According to https://dbmstools.com/categories/database-diagram-tools:
  - **more than 80 tools / only 10 online web-based tools for reverse RDBS engineering, but neither allows the UML-based RDBS representation**.

UML-based RDBS representation

The UML-based RDBS representation has been the subject of research since the beginning of UML development



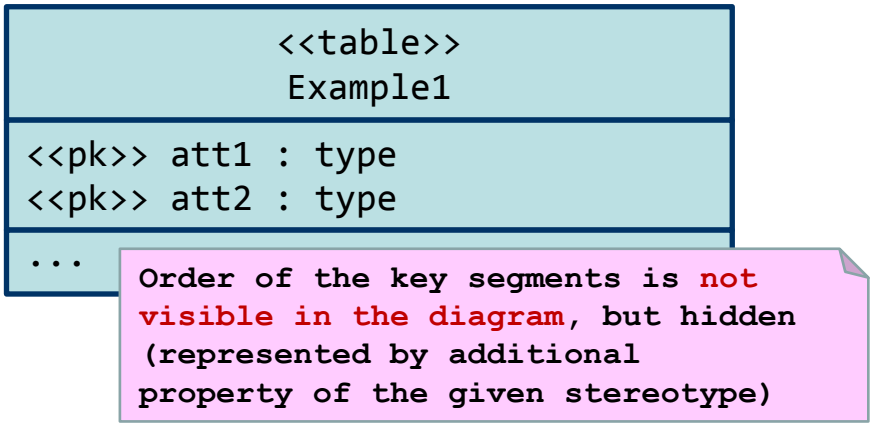
Typical UML stereotypes for RDBS representation
(Naiburg & Maksimchuk)

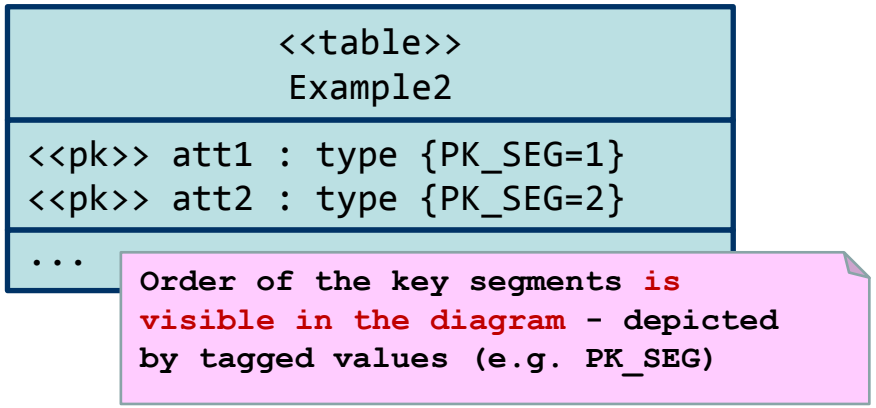


UML-based RDBS representation (cont.)

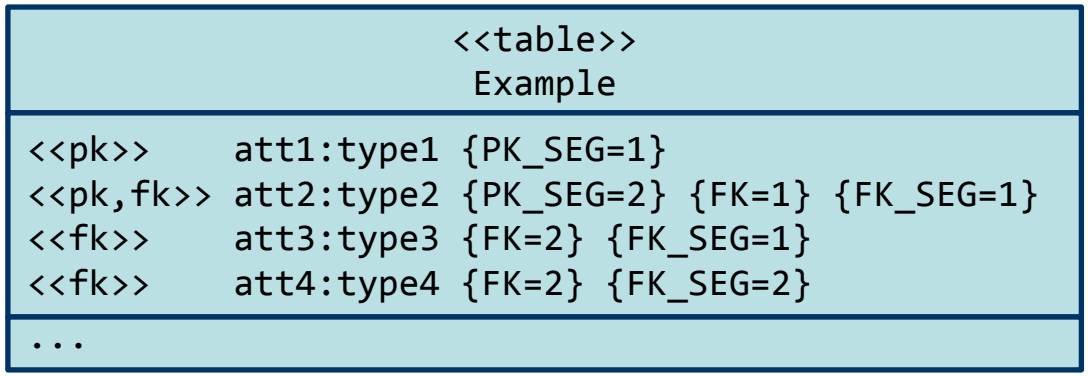
Difficulties with the existing approaches?

- e.g. representation of complex keys





Even more difficulties with the existing approaches?



REDBUL approach to RDBS representation

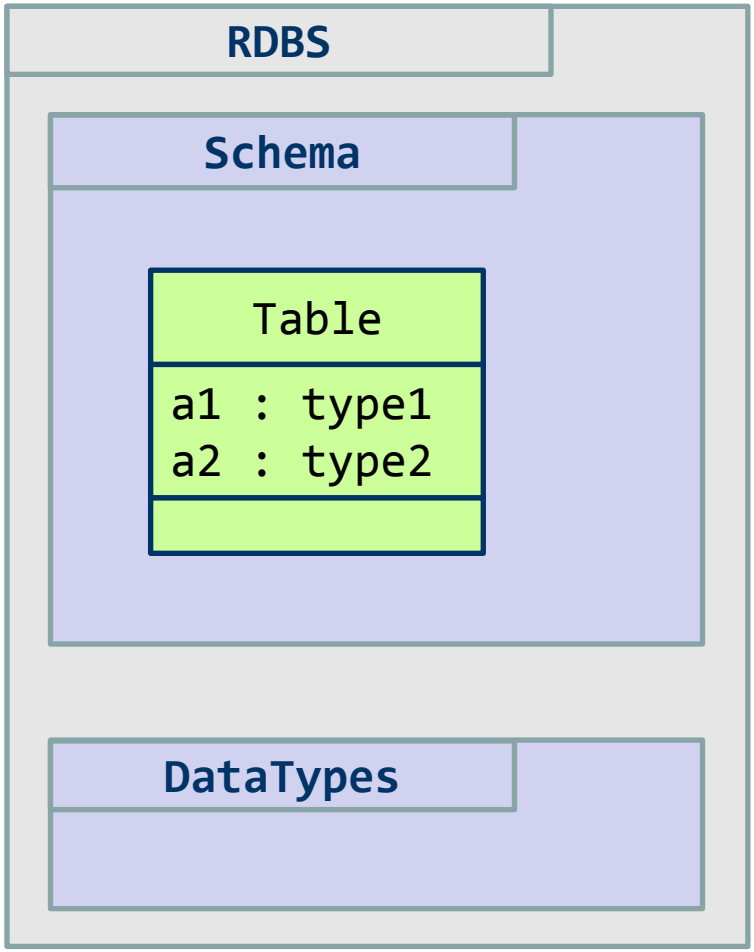
REDBUL = Reverse Engineering of DataBases to UmL

- REDBUL enables database designers to extract the RDBS and visualize it by the standard UML class diagram – **the standard UML notation is used to represent all common RDBS concepts**:
  - RDBS,
  - relational schema/table,
  - primary key,
  - foreign key, referential integrity constraints,
  - index,
  - view.
- Since all concepts are represented by the standard notation, **there is no need to define any domain-specific metamodel extensions** – designers are able to **represent and visualize the RDBS directly in modeling tools**, whereby **no additional tool and/or notation customizations are required**.

REDBUL approach to RDBS representation (cont.)

Relational database schema (RDBS)

- REDBUL represents the entire RDBS by the **same-named UML package**, whereby all RDBS elements are deployed in two sub-packages:
  - **Schema** – contains all RDBS elements except datatypes
  - **DataTypes** – contains datatypes specific for the source DBMS



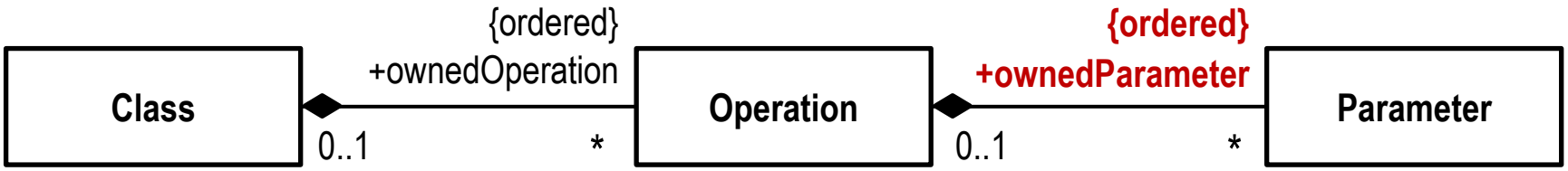
Relational schema (database table)

- REDBUL represents each database table by the corresponding **same-named UML class**, whereby each table column is represented by the same-named class property.

REDBUL approach to RDBS representation (cont.)

Representation of the keys (primary, foreign, …)

- UML possesses an inherent (but not explicit) mechanism that enables very simple and efficient representation of keys.



KEY is an ordered sequence of columns. $key(p_1, \ldots, p_k)$ ⇔ $operation(op_1{:}type, \ldots, op_k{:}type)$ **Operation parameters constitute the sequence.**

KEYS can be modeled by OPERATIONS!
Key segments $p_1,\ldots,p_k$ are represented by corresponding operation parameters $op_1,\ldots,op_k$.

Primary key:
$PK(p_1, \ldots, p_k)$

Foreign key:
$FK(f_1, \ldots, f_k)$

Table
a1 : type1 a2 : type2
PK(a1:type1, a2:type2)

REDBUL approach to RDBS representation (cont.)

Representation of the keys (primary, foreign, …)

Existing approaches

<<table>> Table
<<pk>> a1:t1 {PK_SEG=1} <<pk,fk>> a2:t2 {PK_SEG=2} {FK=1} {FK_SEG=1} <<fk>> a3:t3 {FK=2} {FK_SEG=1} <<fk>> a4:t4 {FK=2} {FK_SEG=2}
...

Our solution based on **the operation signature semantics!**

Table
a1 : t1 a2 : t2 a3 : t3 a4 : t4
PK(a1:t1, a2:t2) FK_1(a2:t2) FK_2(a3:t3, a4:t4)

Direct advantages of our approach:

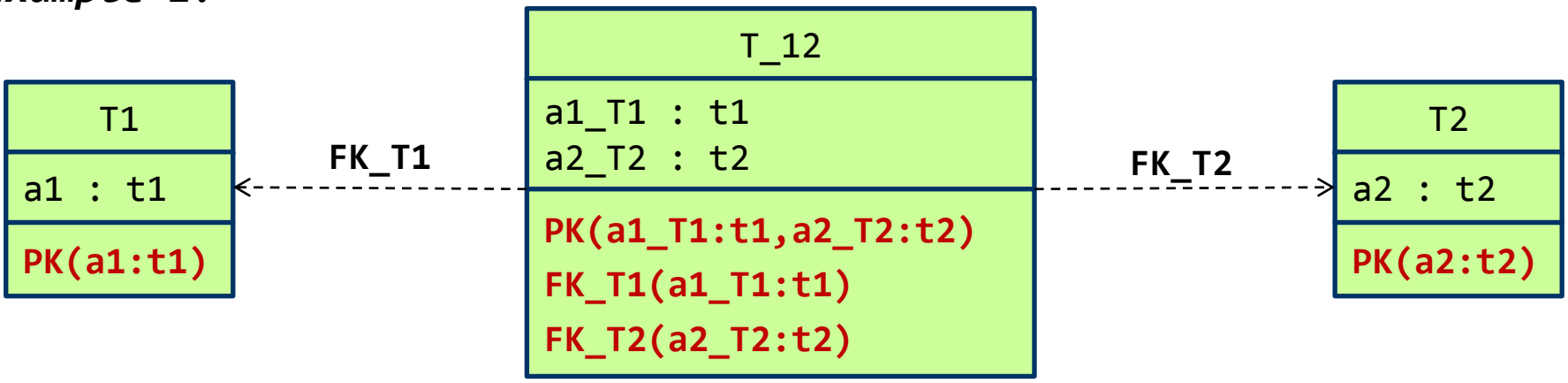
- The main concepts are represented by the **standard UML notation;**
- There is **no need to define and apply some specific profile**;
- **Modeling is easier and faster** than using the specialized notation;
- **Visualization is better** (without specific stereotypes, order of operation parameters represents the key's segments, order of the key segments is visible in the diagram).

REDBUL approach to RDBS representation (cont.)

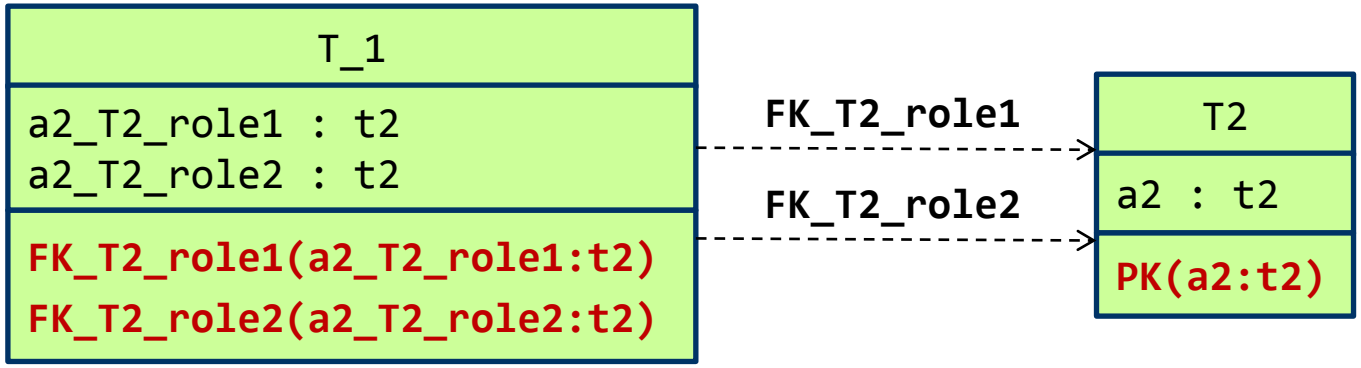
Representation of the keys (primary, foreign, …)

How to deal with multiple foreign keys?

Example 1:



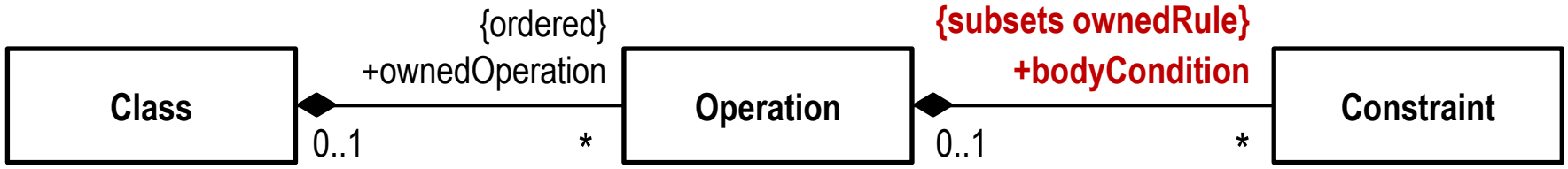
Example 2:



REDBUL approach to RDBS representation (cont.)

Representation of referential integrity actions for foreign keys

- UML allows specification of operation constraints.
- A set of constraints (`ownedRule`) can be specified for each operation.



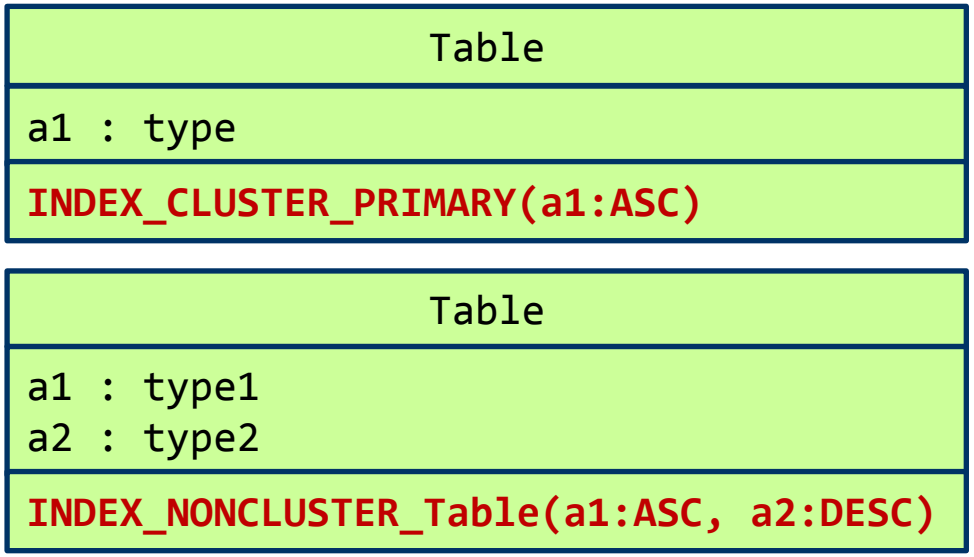
- **In our case, the appropriate constraint specifying the referential integrity actions, is to be defined for each operation representing the foreign key.**
- **The easiest way is to directly specify the DDL statement part**, e.g.

```
ON DELETE RESTRICT ON UPDATE CASCADE
```

REDBUL approach to RDBS representation (cont.)

Representation of indices

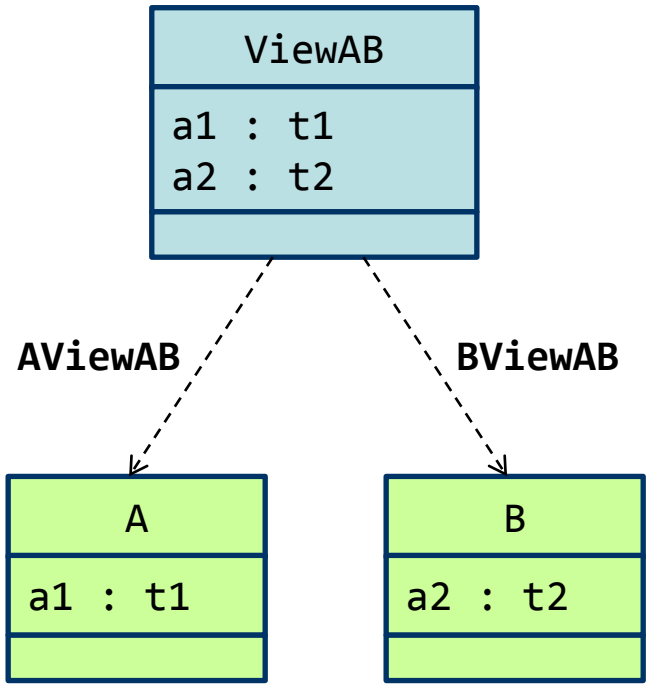
- **Each table index is also represented by an operation in the class that represents the given table**, whereby **the order of the parameters corresponds to the order of the columns in the given index**.
- A **sorting criterion** for each column is **represented by setting the corresponding parameter type to ASC or DESC dummy primitive type** (dataType subpackage).



REDBUL approach to RDBS representation (cont.)

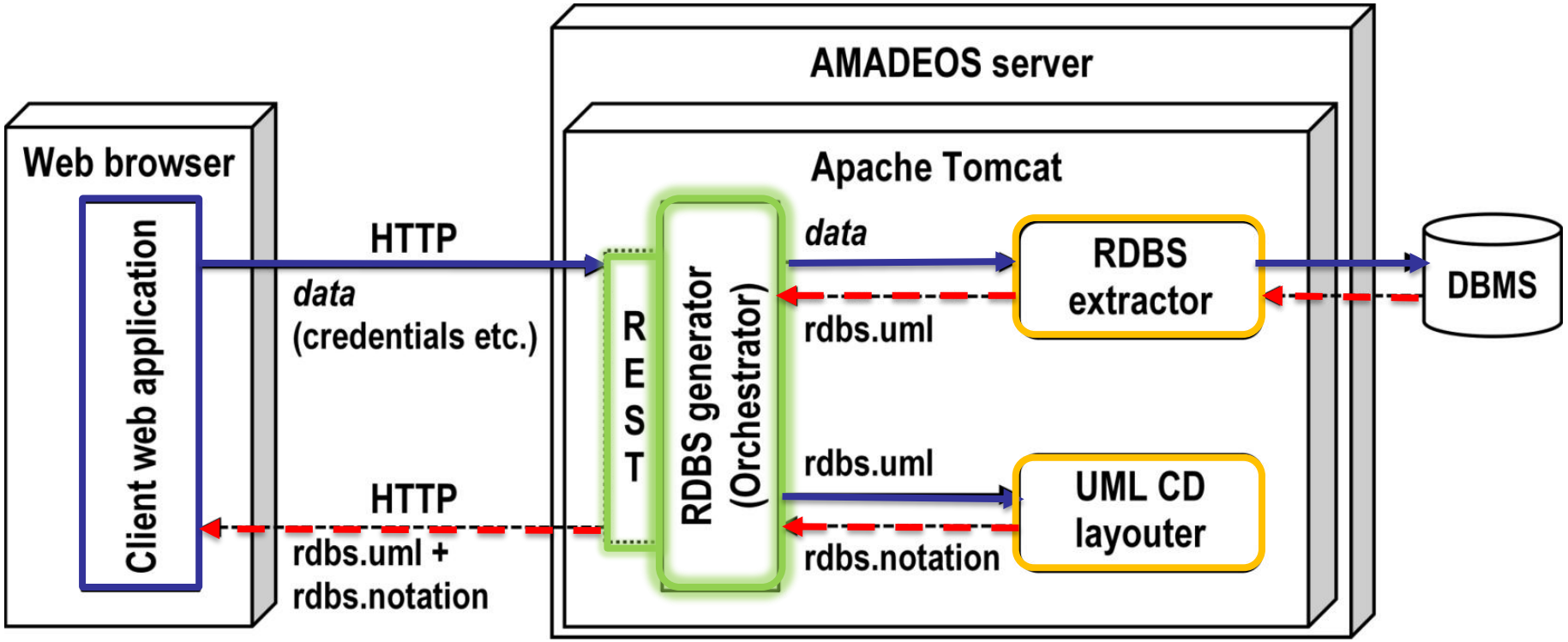
Representation of views

- A **view represents a virtual table** containing the result data set of a stored query referencing other tables and/or views in the database.
- REDBUL represents each view by **the same-named UML class**, whereby **each view attribute is represented by the corresponding same-named class property**.
- **A view dependency on each referenced RDBS element** is represented by the **dependency relationship directed from the class representing the given view towards the class representing the referenced table or view**.



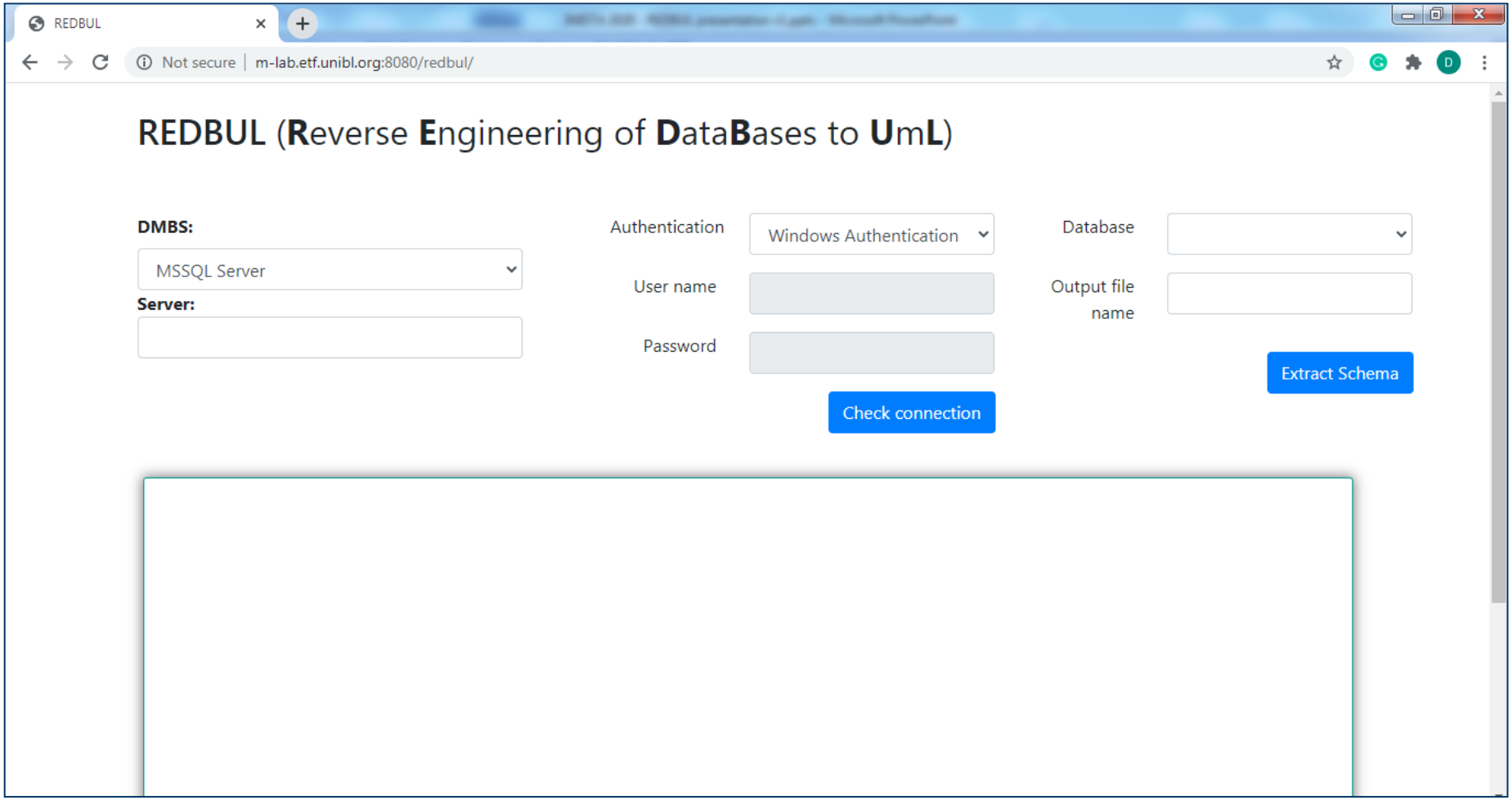
REDBUL system

http://m-lab.etf.unibl.org:8080/redbul/

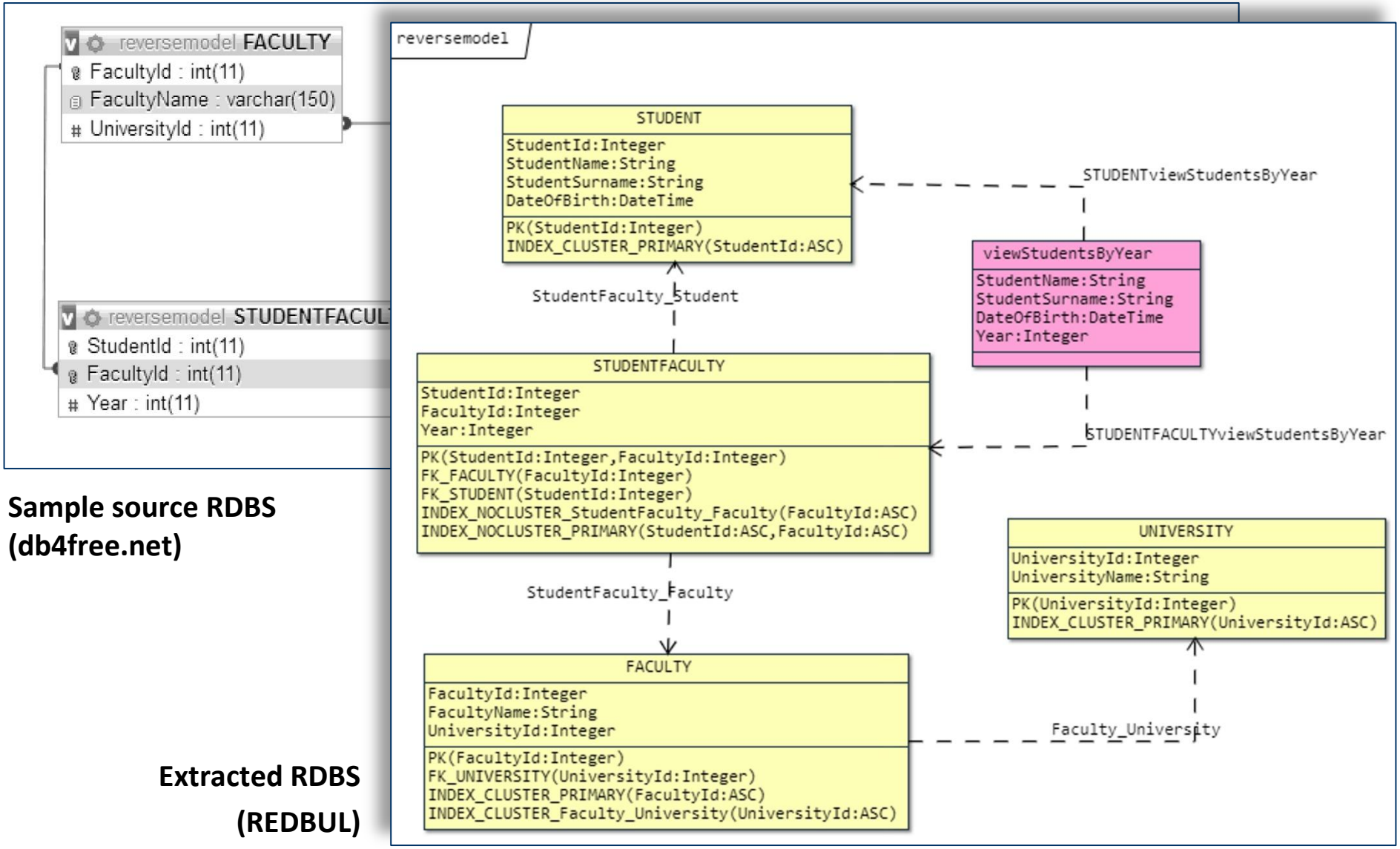


AMADEOS system (cont.)

http://m-lab.etf.unibl.org:8080/redbul/



Illustrative example



Sample source RDBS (db4free.net)

Extracted RDBS (REDBUL)

Conclusion and Future work

- In this paper **we presented an approach to reverse RDBS engineering**, which is completely based on the standard UML notation, **and supported by an online web-based system called REDBUL**.
- In comparison to the existing UML-based approaches, the presented approach and the implemented tool have two main advantages:
  1. The approach is based on the standard notation, which allows database designers to represent and visualize the extracted RDBS directly in the UML modeling tools, with no need to customize the tool or notation;
  2. The implemented REDBUL tool is the first online web-based tool enabling automatic RDBS extraction from an existing database (MS SQL and MySQL) and its UML-based visualization in the web browser.
- **In the future** we plan to:
  - Further **improve the entire approach**;
  - Further **improve the implemented tool**.

INISTA 2020

Innovations in Inteligent SysTems and Applications, INISTA

August 24 – 26 2020, Novi Sad, Serbia

D. Brdjanin, D. Vukovic, G. Banjac,

A. Kelec, I. Dujlovic, N. Obradovic, and D. Banjac

M-lab Research Group @ Faculty of Electrical Engineering

University of Banja Luka, Bosnia & Herzegovina

REDBUL: An Online System for Reverse Engineering of Relational Databases

Thank You!