MANUAL OF ShERML

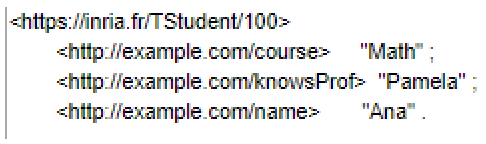
1. INTRODUCTION

ShEx Rule Mapping Language (ShERML) is a tool developed to export data from relational databases to a Resource Description Framework (RDF) graph constrained by a shapes schema. The RDF graph is a data model that describes the semantics of web resources. A shapes schema defines a set of shape names and set of constraints by each shape name. A shapes schema constrains a graph in the number of outgoing edges of a node if the node is typed with a shape name. We call that a node is typed with a shape name if there is a relation over the nodes of the graph and the set of shape names and the node appears in the relation. We allow the constraining of edges to be one (1), at most one (?), at least one (+) and many (*). The shapes schema formalism captures ShEx and SHACL langages, which are languages that constrains the structure of a graph. Consequently, the input for a shapes schema is either a ShEx or SHACL file. The exporting process ends obtaining a graph that satisfies the shapes schema and is ready to be queried by another application.

The beginning step consists of loading a database (sql file) and a shapes schema (json file). Then, the tool shows the relational schema of the database and the shapes schema as class diagrams. Next, the mapping process is done by drawing arrows between bullets that are next to attributes of a table in the relational schema and next to properties of a type name of a shapes schema. Once the set of mappings is defined, the tool materializes a graph from the instance of the relational schema following the mappings. The output format of the graph is in Turtle.

2. SHAPE SCHEMA EXAMPLE

For instance, a graph serialized in Turtle format:



```
<https://inria.fr/TStudent/100>
      <http://example.com/course>     "Math" ;
      <http://example.com/knowsProf>  "Pamela" ;
      <http://example.com/name>       "Ana" .
```

where <https://inria.fr/TStudent/100> is the subject and there are three predicates with its corresponding objects:

- <http://example.com/course> « Math »
- <http://example.com/knowsProf> « Pamela »
- <http://example.com/name> « Ana »

A graph in this context is a set of triples (subject, predicate, object) where the nodes are the subject and objects and the triple itself are the edges.

Now the shapes schema defines the following shapes : TStudent and TTeacher.
The shape TStudent defines the following constraints:
:course :: Literal (1);
:knowsProf :: TTeacher (+)

The shape TTeacher defines the following constraints:
:name :: Literal (1).
where the symbol ¨**:**¨ is the prefix for http://example.com/
Additionally, to this graph we define the following relation over the graph G presented above and a shapes schema that is product of nodes of the graph with shapes names:

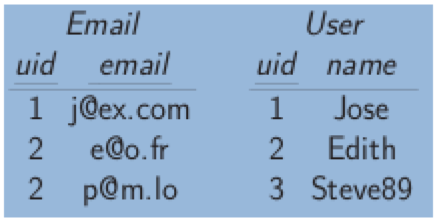
typing(G)={ (<https://inria.fr/Tstudent/100>,TStudent), (¨Pamela¨, Literal), (<http://example.com/course>, ¨Math ¨) }

This graph G satisfies constraint with course but not with knowsProf because the node ¨Pamela¨ is not typed with TTeacher.

3. Mapping Example

1. Relational model

We consider the following database. A database includes its relational model that consists of table name, attribute names of a table name. Additionally, we consider primary keys and foreign keys. Primary key is underlined. For this database, the relational model are tables Email and User. Attribute names of Email are uid and email. Attribute names of User are uid and name. Primary key of Email are uid and email. Primary key of User is uid. There is foreign key from uid of Email to uid of User.



Email		User	
uid	*email*	*uid*	*name*
1	j@ex.com	1	Jose
2	e@o.fr	2	Edith
2	p@m.lo	3	Steve89

2. Shapes schema

Consider the following shapes schema that consists of shape TUser and Lit that represents literal nodes and three triple constraints.

$$\mathsf{TUser} \rightarrow \{:\mathsf{name} :: \mathit{Lit}^{1}, :\mathsf{email} :: \mathit{Lit}^{1}, :\mathsf{phone} :: \mathit{Lit}^{?}\}$$

3. Mapping rules

Consider the two following rules :

- Map name of User to triple constraint composed of property name and target shape Lit.
- Map email of Email to triple constraint composed of property email and target shape Lit.

1. Paths or Join of tables

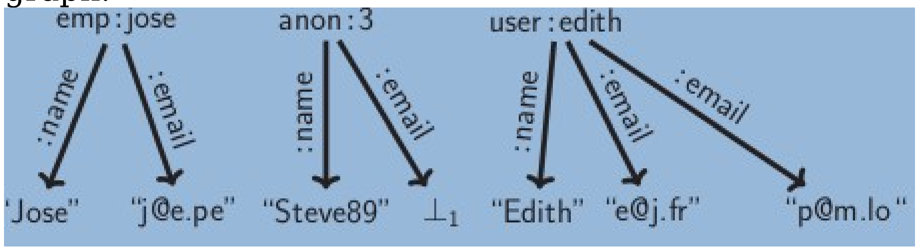
For mapping of name to triple constraint composed of property name, the path is only the table User.
If we have already mapped from User to TUser then for mapping of email to triple constraint composed of property email, the path is Email join User.

2. IRI constructor

Since the data stored in the database are not IRIs then we convert them to IRIs with the help of IRI constructors. For each shape we define a unique IRI constructor. The parameters are primary keys or foreign keys of the table.

4. Solution

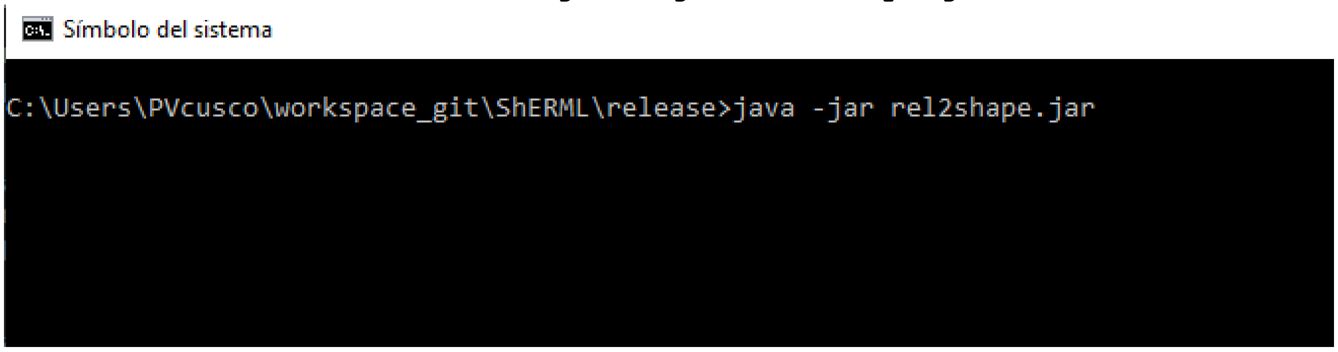
Following the mappings, we will have that the graph is not satisfied. In such case fresh nodes are created to fill the constraints. Therefore, we have the following graph.



4. INSTRUCTIONS

1. Access to the following repository: https://github.com/josemachino/ShERML

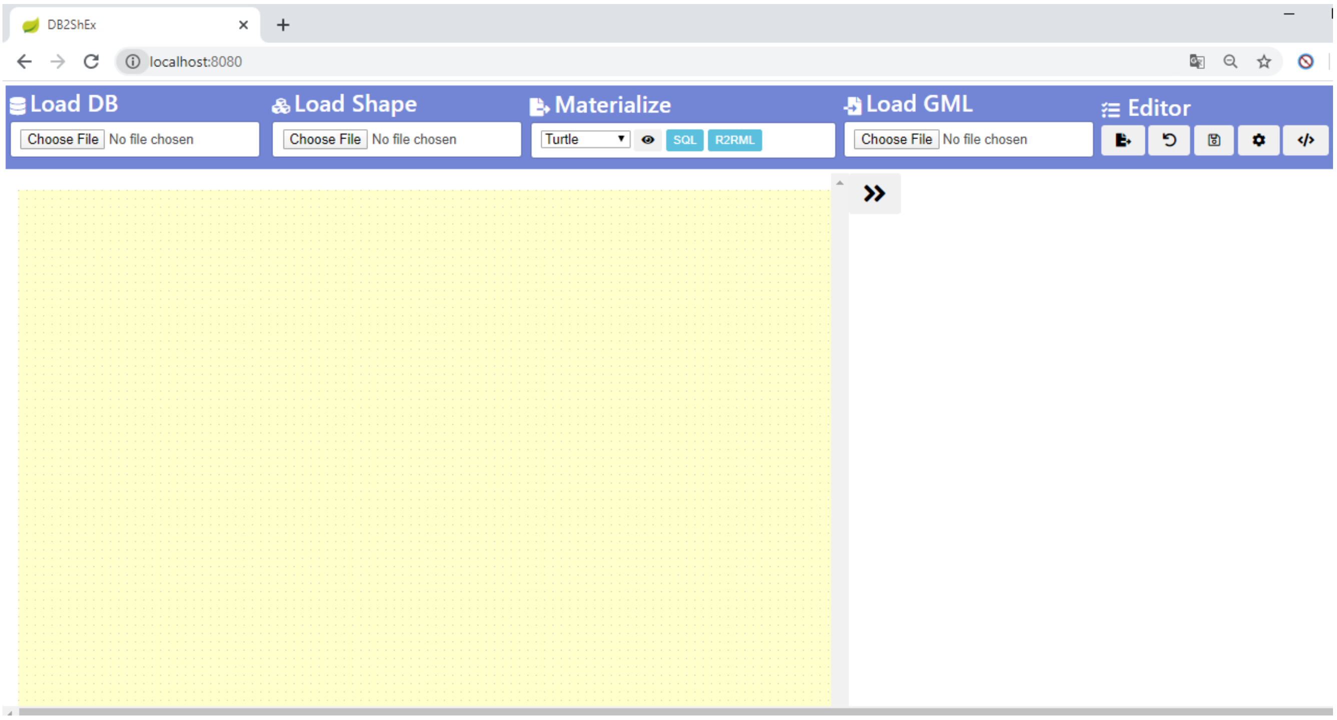
2. On the release folder, donwload rel2shape.jar and launch application from terminal with command : **`$ java -jar rel2shape.jar`**



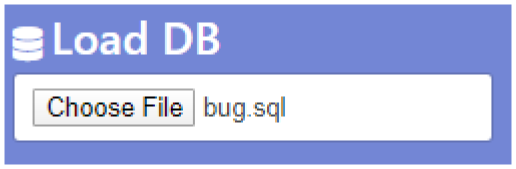
```
Símbolo del sistema

C:\Users\PVcusco\workspace_git\ShERML\release>java -jar rel2shape.jar
```

3. Open an explorer and access to local site : **`localhost:8080`**.The interface of the tool will be shown as the following picture.


DB2ShEx
localhost:8080
Load DB
Choose File No file chosen
Load Shape
Choose File No file chosen
Materialize
Turtle
SQL
R2RML
Load GML
Choose File No file chosen
Editor

4. Open an sql file by clicking in Choose file of Load DB.



5. The system shows the diagram of the schema database. For instance of bug.sql shows the following picture.

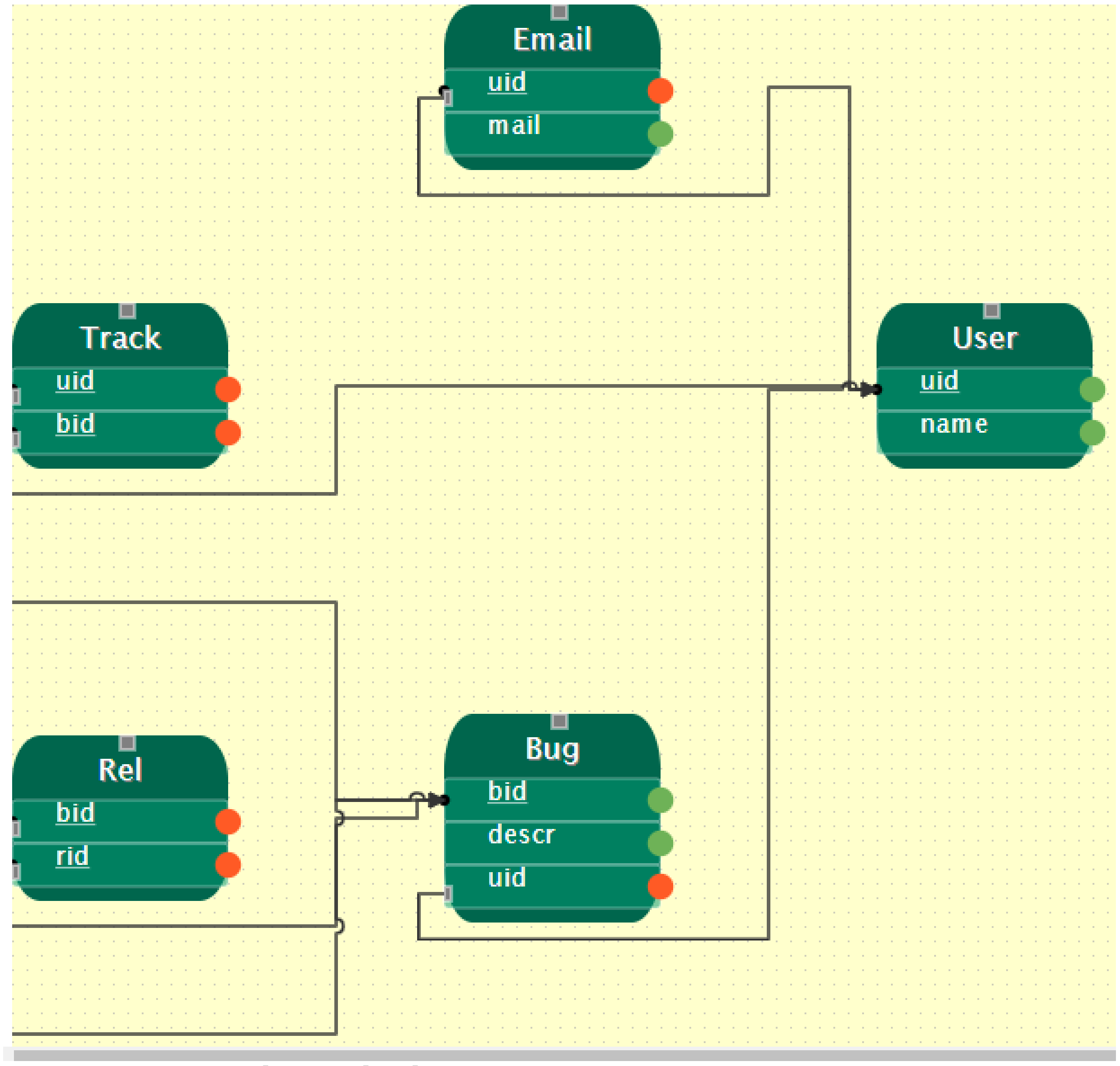
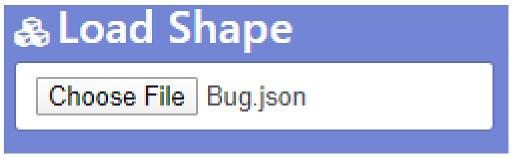


Figure1. Relational schema.

6. Then, open an ShEx or SHACL file by clicking in Choosing file of Load Shape



7. The system shows the diagram of the ShEx or SHACL file as follows.

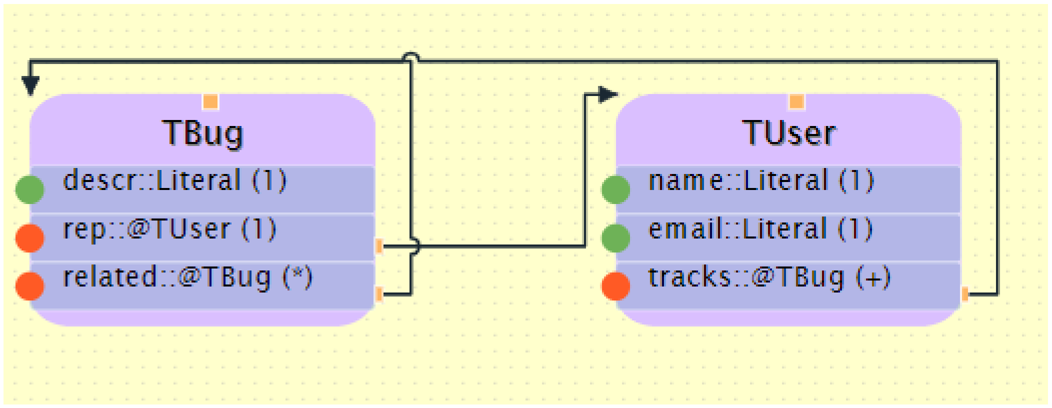


Figure 2. Shapes schema.

We observe that there are two kind of colors in the bullets next to the attributes. In the database (Figure 1), orange color for attributes that are foreign keys and green the rest. In the shape schema (Figure 2), orange color for properties that constraint the target node to be a shape name and green for properties that constraint a literal.

All properties with (1 or +) must be linked because that indicates that the graph must constains an edge with a node whose subject is typed with Tuser and the type of the object is a Literal or Type that is the diagram.

8. Start with the mapping process by dragging from bullets from the relational schema to bullets of the shapes schema.

For instance, we map the following rule:
Map name of User to shape Tuser property name, the result will be Figure 3.

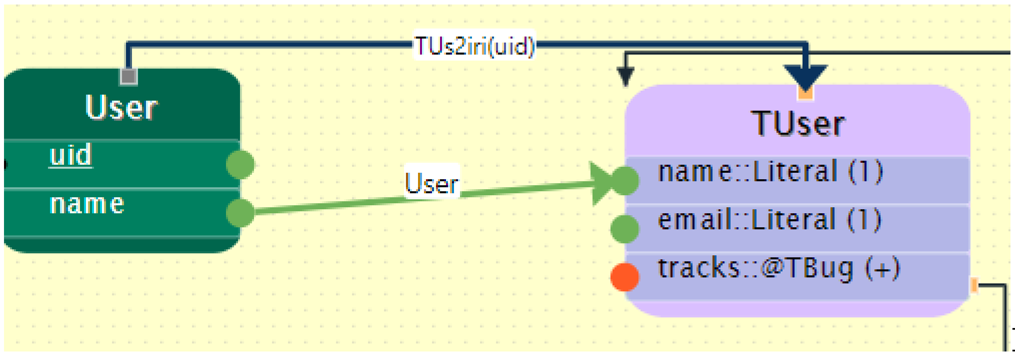
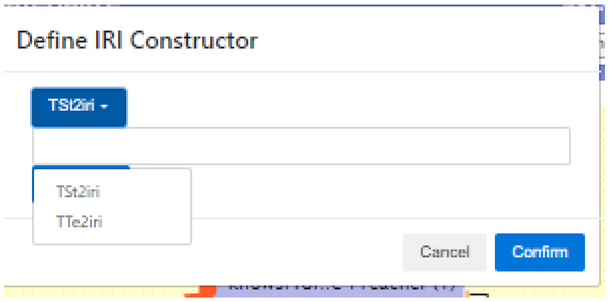


Figure 3.

The user drags from green bullet of name User to green bullet of property Tuser. If the table is related with other tables, the tool asks you to choose the table from where it is connected the main type and the IRI constructor. An IRI constructor is a function that is applied to a set of values coming from the database and returns an IRI. For the mapping process, the tool by default for each shape creates an IRI constructor that is unique to each shape. Therefore, when mapping to constraints that has target type a shape as knowsProf :: TTeacher, user has to select the IRI constructor that corresponds to the target type. For instance, in this message, user select the IRI constructor and the table or join of tables.



The tool by default takes the primary keys of the table to define the parameters of the IRI constructor.

User can see in the right panel of ShERML the mappings done in the editor seen in Figure 4.

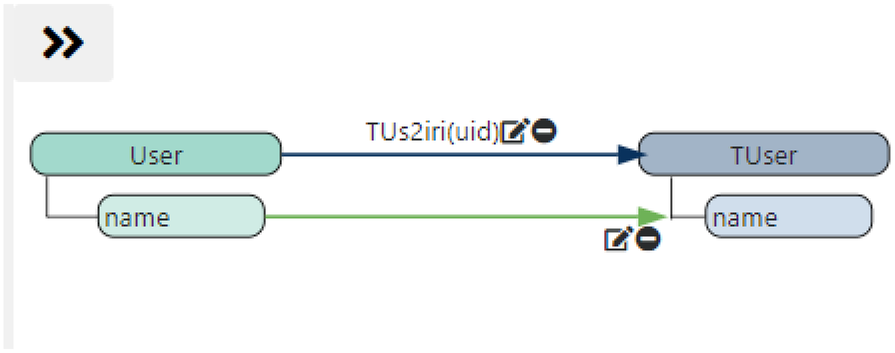


Figure 4.

We can materialize our result by clicking the button of an eye next to Turtle in Figure 5. It is required to have at least a blue arrow and a green arrow. Otherwise, the tool shows a message ¨No database found¨.

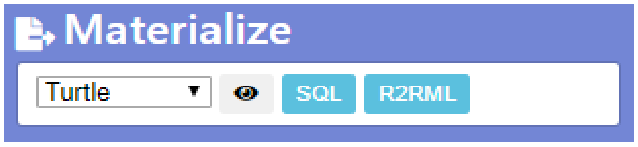
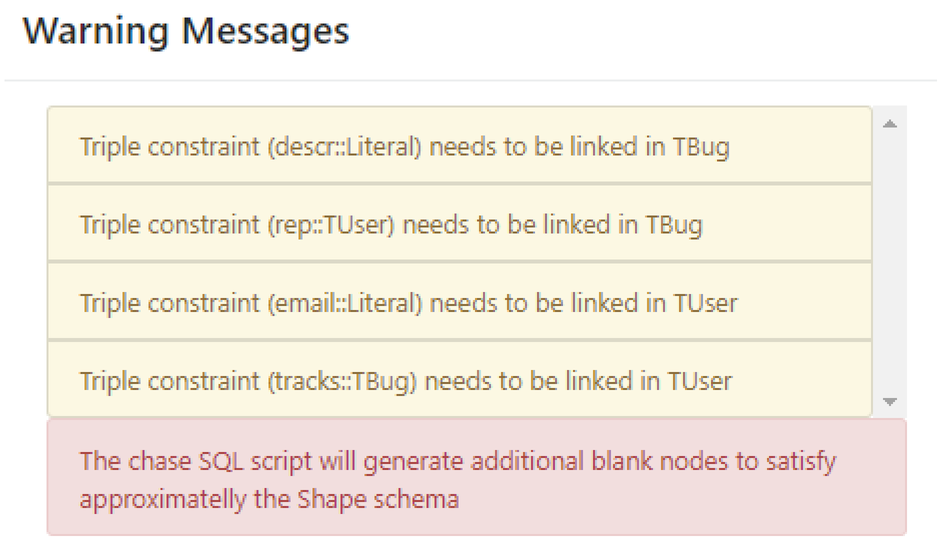


Figure 5.

There are properties that must be linked for multiplicities (1) or (+). If the user does not create mappings to those constraints, then the tool shows the case presented the following message.


Warning Messages

Triple constraint (descr::Literal) needs to be linked in TBug

Triple constraint (rep::TUser) needs to be linked in TBug

Triple constraint (email::Literal) needs to be linked in TUser

Triple constraint (tracks::TBug) needs to be linked in TUser

The chase SQL script will generate additional blank nodes to satisfy approximatelly the Shape schema

The graph resulted for this case will contain values that are identified with @@@ and they represent blank nodes.

Additionally, in the mapping process, when a table is connected with another table, the mapping of the attribute may require the user to precise the path or join of tables such the root table contains the identifier of the node typed with the shape that contains the triple constraint that is being connected. For instance, in Figure 3 suppose there is a table Log that stores error messages tracked by a User with the attributes idLog, idUser and description. If we map idLog to the constraint tracks :: TBug of TUser , tool asks user to select a path (table or join of tables) to know if the tool uses the same value of the primary key of User or another primary key of another table for the shape TUser.