Chapter 4. Database Creation

4.1. Introduction

Database creation is a complex task and involves tuning many parameters. This chapter describes how SAMDSG provides a graphical interface to accept the essential parameters to generate script files, which can be executed from the command prompt to create a new database. This Tool also help us to generate Multidimensional schema generation.

4.1.1 Multidimensional model

A multidimensional model is defined by a fact table and several dimension tables. The first thing that the user has to select from the relational model is the measure attribute. As it has been explained, the measure attribute is the purpose of the cube. It is what the user wants to know, an attribute from the relational model that will be save into the fact table of the multidimensional model. Once the measure attribute is selected the user has to be asked about what is the information the he want to relate with the measure attribute. This information is needed to define the dimensions. Then the grain attribute has to be selected. A grain attribute is selected from the relational model.

Once the grain attribute is selected, the tool has to find a path between the gain attribute and the measure. A grain attribute may have more than one path to the measure attribute. The user chooses the path or the paths that he wants for the multidimensional model. Each path represents a concept of information. Then each couple consisted of path and grain attribute represent different information that the user want to store in the multidimensional model. Therefore for each couple of grain attribute and path, that the user has selected, a new dimension is going to be created. In order to create the fact table are needed the measure and the grain attributes of each selection. It is because the fact table is composed for these elements.
Algorithm Multidimensional_Modelling

Input:
T: { tj } set of tables of the relational model
R: { ( ai , aj ) } set of foreign key relations

Output:
FT = ({g}, {m})
D = {d}
AS = { a }

Variables used in the algorithm
G : variable used to store a grain attribute
m : variable used to store the measure attribute.
P : {pi}: set of path in construction. They start in the grain attribute g.
p : a path.

Begin:
moreGrains : boolean := true;
AS = {}
m := select_measure_attribute_from_relational_model();
// Dimensions
while (moreGrains) loop
g := select_grain_attribute_from_relational_model();
// P : {p} set of path from the measure grain attribute to the measure attribute
P = makePathsGrainToMeasure(T, R, g, m);
for all p ∈ P loop
// d : new dimension
d := makeDimension(T, R, AS, g);
D := D È { d }
P := P È { p }
end loop;
moreGrain := user_answer();
end loop;
// Fact Table
for all d ∈ D loop
The DWH Design Problem Through a Schema Transformation approach

$g' := \text{grain\_attribute\_of}(d);$

$\text{FT} := ((g) \land \{g'\}, \{m\})$

end loop

end algorithm;

4.2. Background

One of the overall goals of this thesis is to be able to create and populate databases for data warehouses. This involves creating a blank database, in which data may be filled in [1]. In this chapter, we discuss how our tool helps towards reaching the first step and second methodologies of the thesis.

4.3. User Interface Design

This section describes how scripts are generated and executed to create a new warehouse database.

All references to ‘Users’ in this section imply the system administrator.

Step 1: Load Application Users may load the tool by entering “java DBHandler” in the command line in the Deployed Application Directory. The main screen of the tool is displayed to users. A snapshot of this screen is presented in Figure 4.1.

Figure 4.1. Snapshot of the main screen
Step 2: Create Database

In the previous step, users loaded the tool. In this step, users may select the ‘Create Database’ option from the ‘Tools’ menu to proceed to the next stage. A snapshot of this screen is presented in Figure 4.2.

Step 3: Edit Parameters

In the previous step, users selected the ‘Create Database’ option. In this step, users are presented with the database creation screen with the default initial values for essential parameters. Users may edit these fields and customize parameters to create a new database. Users may click on ‘Generate Script’ to generate the scripts for database creation. A snapshot of this screen is presented in Figure 4.3.
4.4. Implementation

Component: DBHandler.java
Processing Detail: This is the class that loads the tool. DBHandler creates the desktop with the JDesktopPane and the menu bar using JMenuBar. This desktop displays the internal frames created with JInternalFrame. The menu bar is displayed with ‘Tools’ and ‘Windows’ menu. The ‘Tools’ menu contains options for creating a new database and creating new tables. The ‘Windows’ menu contain the cascading and tiling options for the different internal frames displayed in the desktop.

Users may select one of the options from the ‘Tools’ menu to proceed to the next stage.
Component: DBSwingFrame.java Processing Detail:

This component is called when users select the ‘Create Database’ option in DBHandler.java.
The DWH Design Problem Through a Schema Transformation approach

This component displays the form that users may fill up to generate scripts to create a new database. It uses the Generator_CreateDBFile.java to read from the skeleton files stored in the Skeletons directory and generates the required scripts. The output scripts are written to the output directory specified in the configuration file.

Users may click on the ‘Generate Script’ button to generate the scripts for database creation. Users may execute the newly created batch file to create the new database. A diagram representing the different classes used in creation of a new database is presented in Figure 4.4.

![Diagram of different classes used in creation of a NewDatabase](image)

Figure 4.4. Class diagram representing the different classes used in creation of a NewDatabase.