Design of Tree-Like Database Structure for Solving Test Modeling Tasks of Energy Equipment Training

Authors:
Alexander Gavrilov
Svetlana Blazhenova
Ali Zein
Svetlana Borisova
Andrey Vegera
Ivan Komarov

National Research University “MPEI”

Speacker:
Svetlana Blazhenova
NRU “MPEI”
The Problems Arising in the Design of a New Power Equipment and Their Solution:

Experimental work that is a significant part of the design

Numerical simulation methods (Ansys, Comsol, Star CCM +, etc.)
The Problems Arising During Numerical Modeling:

- Huge opportunities for fine-tuning the solution of the problem
- A huge possibility to set up the task incorrectly
- Deviation of the results from the experiment
The Solution of Above-mentioned Problems:

- A huge possibility to set up the task incorrectly
- Deviation of the results from the experiment

Database containing recommendations for choosing the settings of the grid and the solver module of modeling software
Classical Approach for Database Design

Using classical approach for database design for the task of modeling energy we should:

1) define the main entities
2) define attributes and domains
3) define number of attributes and their values

At this stage, we get one non-normalized relation with all attributes.

Task:
- Physical Process
- Type of Power Equipment
- Place of calculating event

<table>
<thead>
<tr>
<th>Physical Process, Type of Power Equipment, Place of calculating event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydrodynamics, Steam and gas turbines, Nozzle blades</td>
</tr>
<tr>
<td>Hydrodynamics, Steam and gas turbines, Operating Channels</td>
</tr>
<tr>
<td>Hydrodynamics, Steam and gas turbines, Round pipe</td>
</tr>
<tr>
<td>Hydrodynamics, Steam and gas turbines, Confuser</td>
</tr>
<tr>
<td>Hydrodynamics, Steam and gas turbines, Diffuser</td>
</tr>
<tr>
<td>Hydrodynamics, Boiler installations, Convective gas pipeline</td>
</tr>
<tr>
<td>Hydrodynamics, Heat exchangers, Pipe branch</td>
</tr>
<tr>
<td>Hydrodynamics, Heat exchangers, Flow through a perforation in a pipe</td>
</tr>
<tr>
<td>Hydrodynamics, Heat exchangers, Refrigerant inlet</td>
</tr>
<tr>
<td>Hydrodynamics, Heat exchangers, Flat tube diaphragm</td>
</tr>
</tbody>
</table>
Normalizing Database

At the next step, it is necessary to normalize relations and so convert them into the third normal form (3NF). In this way further normalization complicates the database structure and hence the design of the client-server application.

For normalization were created:
- three tables including three components of the task
- table linking these three tables
Disadvantages of the Classic Database Design Approach

However, the study of modern energy systems cannot be limited to a fixed number of attributes. As a result, the static relation will be transformed into dynamic.

The dynamic structure of the database complicates the design of a client-server application:

- collisions between users working simultaneously with the database
- complicated process of ensuring its integrity

A dynamic database structure must contain:

- dynamic relations between parent and child tables
- the ability for changes in data types
Complexities of a Dynamic Structured Database

When moving to a dynamic database structure, it is assumed that the number of attributes, entities and relationships may increase or decrease.

To get data from parent and child tables we can use the join operator (JOIN). For a dynamic structure of the database it will be necessary to dynamically select columns and add join operators.

To get rid of the dynamic structure of the database and convert it to static we replace it with a tree structure.

```
Select t0.*, t1.*, t2.* from dbo.table0 t0 [left] join dbo.table1 t1 on t0.id = t1.id_t0 [left] join dbo.table2 t2 on t1.id = t2.id_t1;
```

```sql
Declare @i smallint -- number of new dynamically generated tables
Declare @k smallint = 0 -- variable used for while cycle
Declare @sql_part0 varchar(max) = ""
Declare @sql_part1 varchar(max) = '"'-- variable for dynamic SQL script
Set @i = 5
Set @sql_part0 = 'Select t'+ convert(varchar(10),@k)+'.*';
Set @sql_part1 = ' from dbo.table' + convert(varchar(10),@k) + ' t' + convert(varchar(10),@k); While @k < @i Begin
Set @sql_part0 = @sql_part0 + ', t' + convert(varchar(10),@k)+'.*';
Set @sql_part1 = @sql_part1 + ' inner join dbo.table' + convert(varchar(10),@k) + ' t' + convert(varchar(10),@k) + ' on t' + convert(varchar(10),@k-1) + '.id = t.' + convert(varchar(10),@k) + '.id_t' + + convert(varchar(10),@k-1);
Set @k = @k + 1;
End
```
Design of Tree-Like Database Structure for Solving Test Modeling Tasks of Energy Equipment Training

Database Design Using Tables with Tree Structure

To get rid of the dynamic structure of the database and convert it to static we replace it with a tree structure.

Advantages of a tree-table:
- Direct access to variables (attributes) and identification of all parent and child variables.
- Transfer of a part of a tree from one position to another.
- Copying tree branches.
- The principle of normalization is not violated.
- There is no need to use dynamic structure and dynamic SQL.
Centralized and Circular Algorithms for Synchronizing Data in a Tree-table

Logical ring representing a circular algorithm:
1) A process goes to the next process, if it is absent then finding the next working
2) If it finds itself hence the circle is passed and data updates

Centralized algorithm with a coordination node:
1) Working processes receive a permission to synchronize from coordination node
2) Then they send reports to the coordinator and go into the standby mode
Implementation of a Centralized Circular Algorithm for Synchronizing Data in a Tree-table

For computers synchronization we use a combined centralized circular algorithm based on both circular and centralized algorithms:

1) Each computer prepares a JSON file
2) They send their own JSON file to the server
3) After merging data from collected JSON files, the previous tree-table is reorganized
4) New generated tree-table data updates in each computer using the circular algorithm.
Conclusion

Tree-like tabular structure in the database allows:

• to realize the tasks of modeling energy processes
• to optimize queries in database
• to create subareas for various physical processes

It is useful for:

• large enterprises that can fill database with their own parameters
• small enterprises that can use existing simulation results
Thank you for attention!

Speaker’s contacts:

Svetlana Blazhenova
NRU “MPEI”
tirend1403fluorit@gmail.com