Hybrid Technique of using ANN in Semi-Star Schema Generation

[1]Aisha Latif, [2]M. Younus Javed, and [1]Ayesha Altaf
College of Telecommunication,
National University of Sciences and Technology (NUST), Pakistan.
College of Electrical and Mechanical Engineering,
National University of Sciences and Technology (NUST),Pakistan.
[1]*aishalatif@mcs.edu.pk*, [2]*myjaved@ceme.edu.pk*, [1]*ayeshaaltaf@mcs.edu.pk*

Abstract

Data warehousing is gaining importance day by day in enterprises, as it helps them to improve their business intelligence. The process of creating a data warehouse needs to be automated so that the transactional sources are generated in least time, with maximum accuracy and with minimum dependability on users. This automation proves its worth particularly when working with small and medium enterprises, where hiring of new people just for creating data warehouses can be unaffordable. The technique presented in this paper automates entity relationship model into data warehouse logical model to generate semi-star schema by using artificial neural networks. More precisely, the step of differentiating dynamic behavior dimensions from static behavior dimensions has been automated by using feedforward back-propagation neural networks. This network ascertains dimensions which are sensitive to changes. The network is trained for all the possible values of inputs and has been tested for actual results. The performance of proposed technique is evaluated by comparing certain metrics like simplicity and minimality with existing data warehouse creation techniques.

1. Introduction

A Data Warehouse (DW) is a huge database that is usually built by integrating several smaller Online Transactional Processing (OLTP) systems [1]. This is done by collecting important facts from each of the OLTP system and transforming it into Online Analytical Processing (OLAP) system. This transformation process is known as Extraction, Transformation and Loading (ETL). ETL itself is a very complicated process, as a result of that, a DW is obtained in which the underlying schema can be star schema, snowflake schema or constellation schema [1,2]. Creating a DW by Semi-Star (SS) modeling schema has an advantage that the OLTP becomes a source of analytical processing [3]. SS schema reduces the cost of ETL process along with saving the cost of creating and maintaining a separate analytical source [4]. For building SS schema, the tables that already exist in OLTP are modified in such a way that they become a source of decision making and this is done by simply adding a few tables into the transactional source. Hence, the DW is created in the existing transactional source.

When talking specifically about automating DWs, a complete system for developing a DW for an enterprise is difficult to design because some steps are dependent on the decisions of enterprise managers. Even then, there is a room for partial automation. Study shows that the process of automating the creation of a DW from Entity Relationship Diagram (ERD) can be easily done by applying some automation steps [8, 10, 12]. But the step of separating Static

Behavior Dimensions (SBDs) and Dynamic Behavior Dimensions (DBDs) is still a challenge, because in star schema the dimensions are not categorized to be either static or dynamic in behavior. But when creating SS schema from an ERD, this step is of great importance [7].

Artificial Neural Networks (ANNs) have been introduced to build such computers and systems which work like human brain [5]. Feedforward back-propagation neural network is a multilayered ANN in which input is received by input layer. This input is then passed on to the hidden layers, from where it reaches the output layer. Error is calculated at the output layer and then weights are adjusted in backward direction. The purpose of using these feedforward back-propagation neural networks is to improve decision making; especially in those cases where the systems are not linearly separable [6].

This paper presents a hybrid technique for creating an SS schema from an ERD by applying some automation steps. The differentiation of SBDs from DBDs is done by involving feedforward back-propagation ANNs. The ANN used in this technique is a three layered network and it uses sigmoid function for activating neurons. Thus, the system automatically detects which dimensions should be static in behavior and vice versa. Consequently, the resulting system generates SS schema in the least time, with maximum accuracy and with minimum dependability on users.

The hybrid technique presented in the paper is then compared with the existing techniques for creating a DW. Comparison is based on the ratio that how much generated schemas are simple and minimal.

Section 2 defines conversion steps from ERD to SS schema. Section 3 presents introduction of feedforward back-propagation neural networks. In section 4, the proposed technique has been explained and section 5 shows results and analysis of technique based on the results. Finally in section 6, the whole work of this paper is concluded.

2. ERD to SS Schema Conversion Steps

ERDs propose the logical model of the data base design of an enterprise. This logical model can be taken as the basis for creating DW logical model. For automating the DW development some steps are needed to be followed. These steps provide a semi-automation technique to generate SS from ERD. This semi-automation requires five steps to be followed to get an SS schema as the output [7]. The sequence of these steps is presented in the following sections.

2.1. Step 1: Normalizing ERD

In this step, the input ERDs are converted into binary ERDs. A binary ERD is one which has all relationships in One-to-Many (1: M) form [8]. To convert an ERD to binary ERD, following steps are taken: All entities which have one to one (1:1) relationship among themselves are merged together. The entities which are connected to each other by Many-to-Many (M: M) relationship are converted into binary by adding a new entity between the connecting entities. This additional entity has (1: M) relationship with both the entities to which it is connected. The ternary relationships are also split by converting them to a series of binary relationships [9]. Figure 1 shows the changes that occur in an ERD after step 1. The resultant ERD after step 1 has all relationships in (1: M) form and such relationships are the most feasible ones to be converted into DW schema.

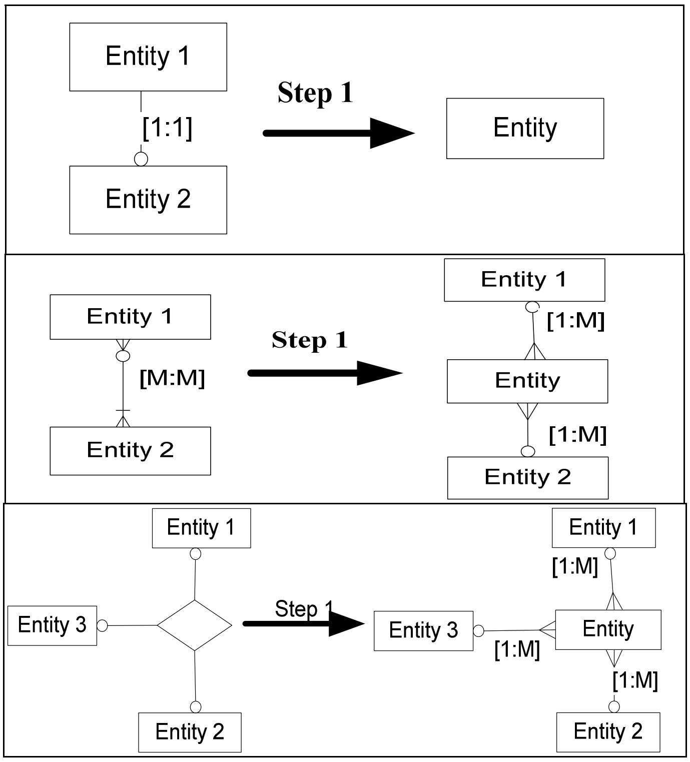

Figure 1. Step 1 on (1:1), (M:M) and ternary relationships.

2.2. Step 2: Differentiating Entities

The output ERD of step 1 is taken as an input for step 2. In this step, the entities need to be classified as transactional, classical or component entities [2]. This is done by calculating Connection Topology Value (CTV) for all entities and determining the threshold value [8]. This threshold is applied for further classification of entities. CTV is calculated by using the following formula:

$$CTV(node) = direct\_w * Count(d\_node) + indirect\_w * \sum CTV(d\_node)$$

Where,

- *CTV (node)* is the connection topology value for a specific *node*
- *direct_w* is the weight assigned to nodes directly connected to *node*
- *indirect_w* is the weight assigned to nodes that are indirectly connected to *node*
- *d_node* is the node directly connected to *node.*

After calculating CTV for all entities, threshold is calculated by putting values in formula 2:

$$T = \text{mean} + k * \sqrt{\frac{\sum_{i=0}^{n} (\text{CTV}(i) - \text{mean})}{n}} \qquad (2)$$

Where,

- T is the threshold value to be calculated
- mean is arithmetic mean of CTVs of all entities
- k is an adjustable variable

- n is the total number of nodes

After the values for CTV and threshold are determined, tables are categorized by applying the following conditions:

```
If CTV>= T then Transaction entity else
    If CTV = 0 and node is not directly connected to transactional entity then
    Classification Entity
     else
     Component entity
```

Here, the transactional entities are candidates of SDTs and component entities are the candidates of conventional dimension tables.

2.3. Step 3: Adding Fact Table

This is the step where fact table is added as an entity to the transactional system. This fact table is connected to all the dimensions except Shared Dimension Tables (SDTs) in the next step. The primary keys of all connected dimensions are added as foreign keys to fact table which makes composite primary key of this fact table.

2.4. Step 4: Joining Dimensions to Fact Table

In this step, component entities are attached to the newly added fact table, with all respective primary keys inserted into the fact table as foreign keys. The dimensions (component tables) added to the fact table have all their relations attached to them (i.e. these component entities are connected to transactional and classification entities).

Time dimension is also joined explicitly to the fact table by adding its primary key in the fact table. The resultant schema after applying step 4 has all the basic features of SS. It contains a fact table and several dimension tables with all their connected relations including SDTs which are shared between dimensions and are not attached directly to the fact table.

2.5. Step 5: Separating SBDs from DBDs

This step is crucial for creating a DW through SS schema because in this step dimensions are categorized as either SBDs or DBDs. SBDs are the dimensions in which contents or schema does not change even after building the DW. But in DBDs, there is possibility for a change to occur in schema or contents of the DW. Changes in a DW can be of two types:

- Changes in data of a DW
- Changes in schema of a DW

Both types of changes, if not handled properly, lead to inconsistency of the DW. To accommodate these changes, some domain experts are required who use their intelligence to define which dimensions can be regarded as dynamic or static in behavior. Mechanization of this step needs to be performed to automate the whole process of creating an SS schema. If whole process is not automated then system is dependent on Data Base Administrator (DBA). Whenever DBA changes, the new comer needs to understand already deployed system from scratch. Any misunderstanding about the work done by the preceding DBA can create problems for the whole system. Therefore, such a system is needed which hardly depends upon user.

Artificial intelligence has been introduced into this system using neural networks. These networks support decision making by taking intelligent decision about determining the behavior of dimensions.

3. Feedforward Back-Propagation ANN

Feedforward back-propagation neural networks are a special type of neural networks in which multiple layers are used. The layers that are embedded between input layer and output layer are the hidden layers [6]. The number of hidden layers may vary from system to system. Input obtained from input layer is passed to output layer through hidden layers.

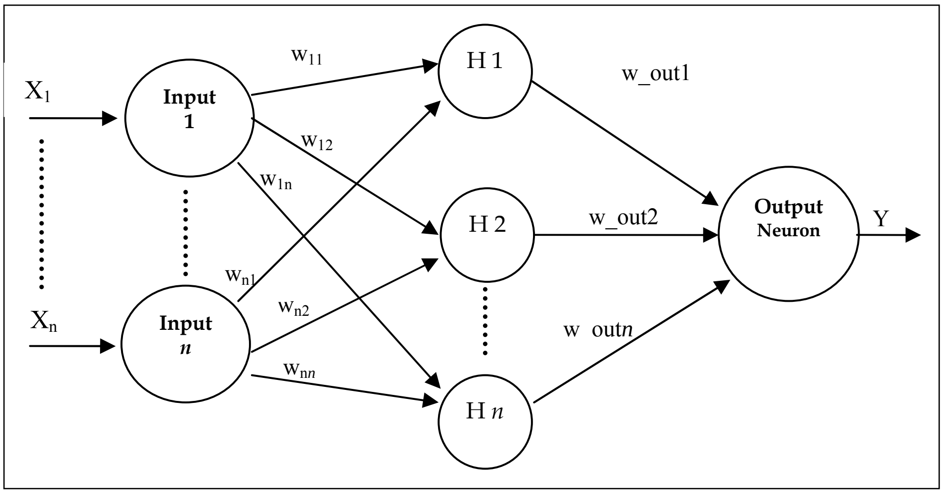

Figure 2. A feedforward back-propagation neural network with a single hidden layer

Connectors between these layers are assigned some weights in the beginning. . Initial weights are assigned in the range: $\left[-\frac{2.4}{n},+\frac{2.4}{n}\right]$ as discussed by Michael Negnevitsky [6].

The activation function used in training of such network is the sigmoid function. Equation 3 defines this function:

$$Y^{\text{sigmoid}} = 1 \Big/ 1+e^{-x} \qquad (3)$$

Figure 2 shows a simple feedforward back-propagation neural network with a single hidden layer, 1 to n neurons in the input layer, n number of neurons in the hidden layer and only 1 neuron in output layer.

The benefit of using this network is that once it is trained, it brings accurate results for those inputs which are very identical to actual inputs.

4. Proposed Technique

The technique presented in this paper involves feedforward back-propagation neural network to automate step 5 defined in section 2.5 in which SBDs are separated from DBDs. The neural network created in this proposed technique has three neurons in the input layer, four neurons in hidden layer and a single neuron in the output layer. The input for this network is obtained from three sources:

- Type of the entity
- Type of change
- CTV calculated in section 2.2

Only those CTV values are taken as input for this network which are below threshold value (i.e. all classical and component entities and no transactional entity). Type of the entity describes whether that particular entity is a controllable entity for that enterprise or uncontrollable. The input for determining an entity as controllable or uncontrollable is acquired from user. Furthermore, changes are categorized as data changes or schema changes in controllable entities and in uncontrollable as well.

Thus, the input-output table for this neural network is like Table 1. According to this table, if value of first input is '1', it means that the dimension is under control of respective enterprise and if this value is '0', then it is uncontrollable for that enterprise. Second input shows the type of change to be either data change or schema change. If this input is '1', there are chances for data to change and for possibility of changes to occur in DW schema; its value is '0'. Third input has value '0' if CTV is '0' for a particular dimension and '1' if CTV is greater than 0. In output column, '0' shows that the dimension is static in behavior and '1' shows that the dimension is a DBD. On the basis of this available data, the proposed system is designed. The proposed neural network looks like the one shown in Figure 3.

Table 1. Inputs and output for the neural network.

S. No.	Type of Entity	Type of Change	CTV	Output
1.	1	1	1	0
2.	1	1	0	0
3.	1	0	1	0
4.	1	0	0	1
5.	0	1	1	1
6.	0	1	0	0
7.	0	0	1	1
8.	0	0	0	1

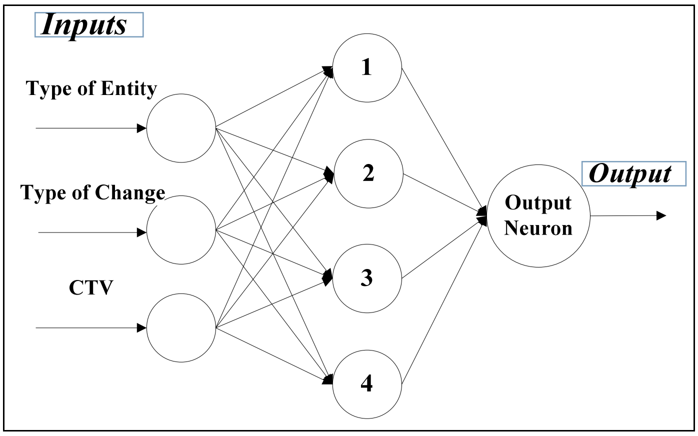

Figure 3. Structure of feedforward ANN built for the proposed technique

This proposed technique has been implemented in MATLAB. Initially, weights are assigned values in the range defined in section 3. The activation function used for this neural network is sigmoid function mentioned in equation 3. Value of sum of squared errors has been kept lower than usual (i.e. at usual it is taken as 0.001 as mentioned in [6]) but in our proposed technique, sum of squared errors is equal to 0.0001 in order to make this system more accurate.

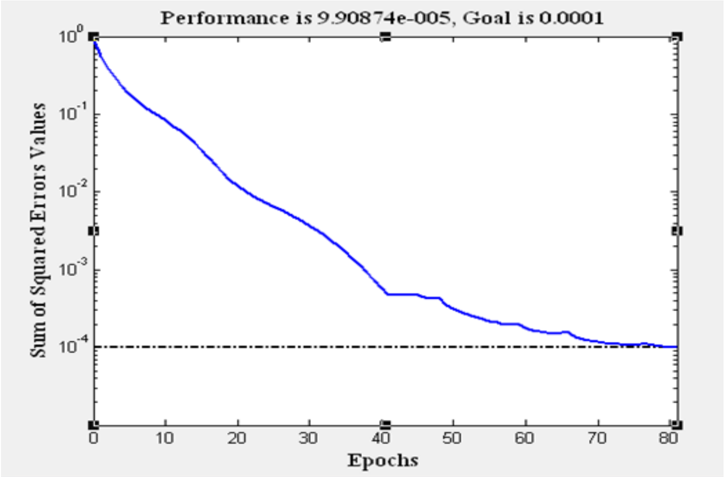

Figure 4. Training curve of ANN with learning rate (dotted line shows goal and solid line shows training)

Performance was observed by varying learning rate α between ‘0.150’ to ‘0.195’. By taking values of α to be 0.195, 0.185, 0.175, 0.165 and 0.15, the performance goal was achieved after 592, 81, 96, 358 and 282 epochs respectively. Thus the performance goal is met in least epochs when value of learning rate was kept ‘0.185’. Figure 4 shows the training curve of ANN for the selected variable values for the proposed technique when the goal has been met in 81 epochs.

After training the system, testing was performed. It was done by entering values for all three inputs. Outputs obtained from these inputs were then tallied from actual outputs, keeping sum of errors in focus for using sigmoid function.

After the system identifies a dimension to be SBD or DBD, surrogate keys are added to these DBDs in addition to existing primary keys. The foreign keys already present in SDTs and fact table are replaced by these newly added surrogate keys as new foreign keys.

For explanation of the proposed technique, an example is presented. In this example, inputs are supposed to be:

- Entity is uncontrollable, so the value for type of entity column is ‘0’.
- Type of change is data change, so its column value is ‘1’.
- CTV > 0, so it appears as ‘1’ in input column.

When these inputs were entered into the system, it displayed output value to be ‘0.9951’. This shows that the dimension is a DBD. This output value can be tallied from the output value in row 5 of Table 1, where its value is ‘1’. Similarly, other inputs can be entered into the system to see obtained output from the proposed system.

As it can be observed from above example, the DBDs are separated from SBDs by taking previously calculated CTV along with some input from user regarding the type of entity and type of change. Therefore, system is working intelligently to make decisions for this step.

5. Results and Analysis

Previous section describes the proposed technique, which was evaluated by comparing simplicity and minimality metrics of resultant SS schema.

Different formulas have been designed by various researchers to evaluate simplicity and minimality in a data warehouse schema [11]. Simplicity ensures that the schema contains least possible constructs. So, a schema is said to be simple if it contains more entities and less relationships. By minimality, it is meant that the schema contains information of all facets only once, consequently to ensure that the schema is least redundant. A DW is complete if it contains all important features of application area.

For evaluating the proposed technique, 15 case studies have been taken from different resources. These resources include internet and several other studies. ERDs of these case studies are converted into DW star schema by applying existing techniques. Out of all existing approaches, only two are used to compare the proposed technique, and those are SAMSTAR [8] and Schema-transformation approach [10]. Both these approaches are not inelligent as they do not involve any artificial intelligence. After converting ERDs of all case studies into star schema by all three approaches, above mentioned metrics are calculated for resultant schemas.

The first metric that was compared is simplicity [11]. The formula used to calculate simplicity of a schema is:

$$\textbf{Simplicity} = \frac{\mathbf{Ne}}{\mathbf{N_e} + \mathbf{N_h} + \mathbf{N_r}} \tag{4}$$

Where,

N_e is the number of entities, N_h is number of inheritance links and N_r is the total number of relationships in a schema.

Table 2 shows the values of simplicity for SAMSTAR, schema transformation and the proposed technique. These values show the percentage of simplicity found in each schema and a comparison to other techniques can be made from the proposed approach.

Table 2. Values of Simplicity metric

CASE STUDY NO.	**SCHEMA-TRANSFORMATION**	**SAMSTAR**	**PROPOSED TECHNIQUE**
1.	0.65	0.8	0.75
2.	0.65	0.75	0.85
3.	0.79	0.77	0.87
4.	0.9	1	0.98
5.	0.65	0.89	0.9
6.	0.55	0.87	0.875
7.	0.78	0.97	1
8.	0.66	0.76	0.88
9.	0.9	0.88	0.79
10.	0.85	0.85	0.78
11.	0.88	1	0.98
12.	0.765	0.88	1
13.	0.65	0.85	0.875
14.	0.56	0.77	0.8
15.	0.7	0.78	0.9

From above table, it can be easily observed that the results of the proposed technique are equivalent to existing ones and even in some cases, it gives better results than the existing approaches.

The second metric to be calculated was minimality. This metric ensures that the resultant schema is least redundant. Formula to calculate minimality [11] in a schema is given in equation 5:

$$\textbf{Minimality} = \frac{\sum_{n} w_i N_{type\_i} - w_i N_{rtype\_i}}{\sum_{n} w_i N_{type\_i}} \qquad (5)$$

Where,

- Ntype_i is number of elements of type (class, inheritance link, association link)
- Nrtype_i is the number of redundant elements of a type
- wi is the weight assigned to a specific type

For all the 15 case studies, ERDs are first converted into DW schemas by applying the respective approaches. Then from the resultant DW schema, minimality is calculated. Figure 6 shows values of three techniques which are compared that include Schema-Transformation, SAMSTAR and proposed technique.

It can be easily observed from Figure 6 that the proposed technique gives minimality values for all schemas ranging from 0.80 to 1. The minimality values for schema transformation approach never reaches to 1 (i.e. the resultant DW schemas generated after applying this approach does not completely remove redundancy). One of its reasons can be, that the application of transformation steps is done manually and the decision is taken solely by the developer.

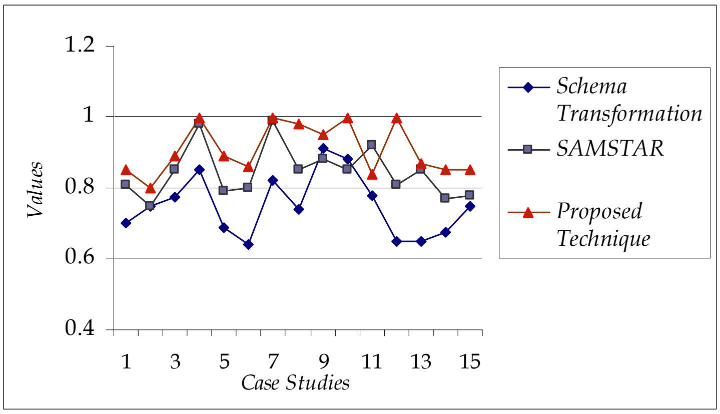

Figure 5. Minimality values obtained from different approaches

In the second technique, the SAMSTAR, results are much closer to the value 1, which shows that the generated schemas are less redundant and the technique is good to be applied. But this technique generates star schema and same technique cannot be applied to generate semi-star schema, where static dimensions are different from dynamic ones.

The proposed technique generates semi-star schema and it is obvious from the graph that the schemas generated from this technique are less redundant and are least user-dependant.

6. Conclusion

A hybrid technique has been proposed to make decisions for creating a data warehouse using neural networks with underlying schema as SS schema. Neural network used in the proposed technique is feedforward back-propagation neural network. Five steps are presented to convert an ERD to SS schema and the involvement of neural networks has only been introduced during the last step in order to take intelligent decisions for determining the behavior of dimensions. The ANN used divides dimensions into two categories (i.e. SBDs and DBDs). The values of input, output and learning rate are all defined in the proposed technique. Thus, the process of generating SS schema from an ERD is simplified with lesser user involvement, minimum time and better accuracy.

The proposed technique is then compared with the existing techniques that do not involve any intelligence. A comparison criterion is based on simplicity and minimality metrics. Results show that the proposed technique is comparable to the existing ones, and in some cases, it gives better results than previous techniques. In future, this proposed technique can, therefore, be applied in a fully automated system, with all steps being performed automatically. Such systems will be able to take an ERD and convert it into SS schema without any involvement of the user.

References

[1] Kimball, R., and Ross, M., "The Data Warehousing Toolkit", John Willely & Sons, Inc., 2002.

[2] Daniel L. Moody and Mark A.R. Kortink. "From Enterprise Model to Dimensional Model: A methodology for Data warehouse and Data Mart Design". Proceedings of 2nd International Workshop on DMDW, Stockholm, Sweden, June 5-6, 2000, Pg 5-1 – 5-12.

[3] M. A. Pasha, J. A. Nasir and M. K. Shahzad, "Semi-Star Modeling Schema for Managing Data Warehouse Consistency". Proceedings of NCET, Karachi, Pakistan, December 18, 2004, pp 63-66.

[4] Khurram Shehzad, "Semi-star Schema for Operational and Analytical Requirements of SMEs", International Journal of Management and Decision Making (IJMDM): Special Issue on "Decision Support System and Knowledge Management in SME's", Greece, 2008.

[5] M. Gupta, J. Szymanski, A. Sharma, and R.K. Soni, "Human Vibration Monitoring in Large Haul Trucks - An Artificial Neural Network Approach", In Proceedings of Applied Simulation and Modeling 2008

[6] Michael Negnevitsky, "Artificial Intelligence, A Guide to Intelligent System" Addision-Weslay, Pearson Education 2002.

[7] Latif, A., Younus Javed, M. and Khattak N. S., "Intelligent Technique to Determine Behavior of Dimension Tables in Semi-Star Schema Generation", N.S. International Conference on Convergence and Hybrid Information Technology, 2008. ICHIT '08, 28-30 August, 2008, pp 389-393.

[8] Il-Yeol Song, Ritu Khare and Bing Dai. "SAMSTAR: A Semi-Automated Lexical Method for Generating Star Schemas from an Entity-Relationship Diagram". In Proceedings of the ACM Tenth international Workshop on Data Warehousing and OLAP, DOLAP '07, ACM, Lisbon, Portugal, November 09, 2007, pp 9-16.

[9] Il-Yeol Song and Jones, T.H. "Ternary Relationship Decomposition Strategies Based on Binary Imposition Rules". In Proceedings of the Eleventh International Conference on Data Engineering, IEEE Computer Society, Washington, DC, March 06 - 10, 1995, pp 485-492.

[10] Marotta A. and Ruggia R., "Data Warehousing Design: A Schema-transformation Approach", In proceedings of 22nd International Conference of the Chilean Computer Science Society, 2002. SCCC 2002, IEEE Computer Society, Atacama, Chile, 6-8 November 2002, pp 153- 161.

[11] S. S. Cherfi, J. Akoka and I. Comyn-Wattiau Conceptual Modeling Quality - From EER to UML Schemas Evaluation, ER 2002, LNCS 2503, Springer Berlin / Heidelberg, 2002, pp 414-428.

[12] Latif, A.; Younus Javed, M. and Sharifullah K., "Semi-Automated Approach for Converting ERD to Semi-Star Schema"4th IEEE International Conference on Emerging Technologies, ICET '08, IEEE, Rawalpindi, Pakistan, 18-19 October 2008, pp 264-268.

Authors

Aisha Latif received her BS degree in Computer Sciences from COMSATS Institute of Information Technology, Lahore, Pakistan in 2006. Nowadays she is a student of College of Telecommunication (MCS), NUST, Rawalpindi, Pakistan. She has been doing her MS in Computer Software Engineering with major in data warehousing. Her areas of interest are database systems, data warehousing, artificial intelligence and design and analysis of algorithms. She has 2 international publications to her credit.

Dr Muhammad Younus Javed did his PhD in Adaptive Communication Systems from University of Dundee, Scotland, United Kingdom in 1991 and MS in Predictive Systems from the same university in 1988. He completed BE Electrical Engineering from UET Lahore, Pakistan, in 1982. He is serving in the College of Electrical and Mechanical Engineering (CE&ME) since 1991 and has taught a number of courses at undergraduate and postgraduate level. He is currently Head of the Computer Engineering Department at CE&ME, National University of Sciences & Technology (NUST), Rawalpindi, Pakistan. His areas of interest are biometrics, parallel systems, operating systems, computer networks, digital image processing, database systems and design & application of algorithms. He has 146 national/international publications to his credit.

Ayesha Altaf is doing her MS in Information Security from College of Telecommunication (MCS), NUST, Rawalpindi , Pakistan. She did her BS Computer Sciences from COMSATS Institute of Information Technology, Lahore, Pakistan in 2006. Her areas of interest are data communication, network security, artificial intelligence and algorithms. She has 3 international publications to her credit.