ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2015

Generalized Algorithm for the Proposed Database Model

Ranjana Ingolikar[1], Rasika Khandal[2], Rahul Mohare[3]

HOD, Dept. of Computer Science, SFS College, RTM Nagpur University, Nagpur, India[1]

Assistant Professor, Dept. of MCA, SRPCE, RTM Nagpur University, Nagpur, India[2]

Assistant Professor, Dept. of MBA, DMIMS, RTM Nagpur University, Nagpur, India[3]

ABSTRACT: We have proposed a new algorithm for selecting the better database model for satisfying the user's requirement. We have already done the parameterized comparative study between different database models in our research study. The comparative database models are HDBMS, NDBMS, RDBMS and OODBMS. The comparative study has done on the basis of 14 selected parameters. Some of the Core parameters and remaining are Performing parameters. In this paper we have proposed the generalized algorithm which helps to find out best data model to the user. The algorithm is executed in .net environment and is especially designed for the user to accept his requirements.

KEYWORDS: DBMS, Defining Weights, Defining Anchors, Dashboard, Generalized Algorithm

I. INTRODUCTION

A database is storage of inter-related persistent data. The database systems give us a provision to store the data and management of these database systems include giving us a possibility to manipulate and access the data through various database functions like Create, Retrieve, Update and Delete and also providing structures so that we can store the persistent data. The data should be persisted in a way that it with stands system crashes, provides security against the potentially harmful users and anomalous data should not be stored when the database is being accessed by multiple users.There are a variety of database management systems categorized on the database models. They are as follows:

1. Hierarchical data model
2. Network data model
3. Relational data model
4. Object oriented data model

In our previous study we have done parameterized comparison of different database management system models that has been considered in our study. The considered models are HDBMS, NDBMS, RDBMS and OODBMS. The comparison has been done in 13 parameters these parameters are listed in fig.1.

In this paper we have done an exploratory study to justify the selection of the parameters used for comparison and has built the groundwork for the proposed model of selection of database model. We have considered two basic parameters for comparison of these models. The parameters so selected are broadly categorized in to two different groups namely Core parameters and Performing parameters. The Core parameters are basic DBMS parameters commonly supported by all database models. The Performing parameters are selected DBMS parameters in which their functionalities are varying in each of the database model.
The core and performing parameters comprises of following selective parameters for comparison, which is presented in the following diagram:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2015

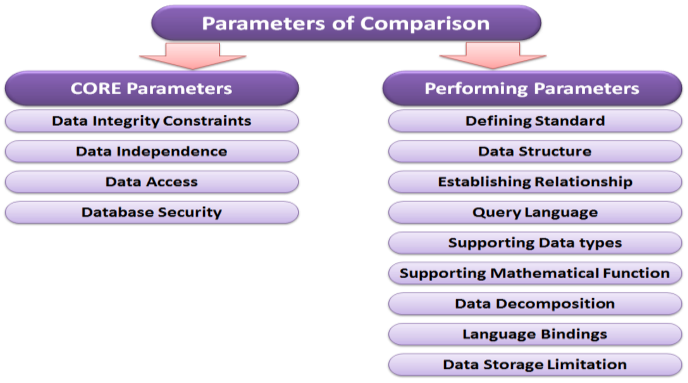

Fig. 1Parameters of Comparison

The basic idea is to propose a conceptual framework which enables the users of DBMS models to select the best DBMS model to store the responses of the questionnaire, if any one wants to opt for DBMS models.

A. *Assigning weightage to parameters*

Here we have proposed a method of selecting best database model for any type of questionnaire. In this method we have assigned some weights to each of the parameters. The weights so assigned shall sum up to 100. These assigned weights will be useful for calculating the data model index.

- Core parameters are assigned equal weightage (7 each)
- Performing parameters are assigned equal weightage (8 each)

In all we have 4 core parameters and 9 performing parameters. The maximum value of the performance index comes out to be 100 [7*4=28 + 9*8=72 ==100].

We have gone through the simplest logic that the models which have best functions and facilities build among them shall have more weights. On the basis of this assumption we have assigned 8 points to the performing parameters and these are floating weightage (deemed to be different, depending on the functionality) and those parameters which are found in all database models and those which are considered to be of equal importance through all the database models are also given equal weightage, but these are constant weightage i.e.7 points. Which can be better understood from the following simple diagram:

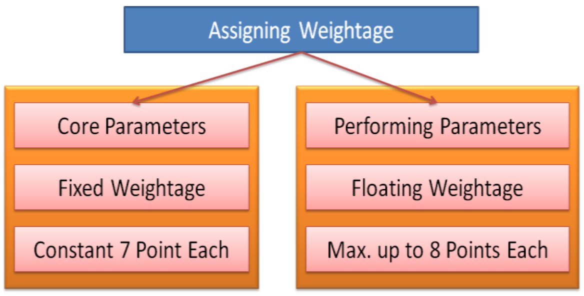

Fig. 2 Assigning Weightage

Following TABLE I represent the assigned weightage to 13 selected parameters those are listed in Fig. 1.

Sr. No.	Considered Parameters	Assigned weight
	Core Parameters	
1	Data Integrity Constraints	7
2	Data Independence	7
3	Data Access	7
4	Database Security	7
	Performing Parameters	
5	Defining Standard	8

6	Data Structure	8
7	Establishing Relationship	8
8	Query Language	8
9	Supporting Data Types	8
10	Supporting Mathematical Functions	8
11	Data Decomposition	8
12	Language Binding	8
13	Data Storage Limitations	8
	Sum of assigned weight	**100**

TABLE I Assigned weightage to 13 parameters

The varying weightage assigned to the performing parameters depend on the functionality and usability present in that particular data model. The assignment of weightage to each of the database model is important in calculating of data model index. Following TABLE II shows the overall weightage assigned to database models.

Sr. No.	**Considered Parameters**	**Assigned Weightage to DBMS Models**			
		HDBMS	**NDBMS**	**RDBMS**	**OODBMS**
CORE Parameters					
1	Data Integrity Constraints	7	7	7	7
2	Data Independence	7	7	7	7
3	Data Access	7	7	7	7
4	Database Security	7	7	7	7
Performing Parameters					
5	Defining Standard	8	8	8	8
6	Data Structure	6	5	8	7
7	Establishing Relationship	5	6	7	8
8	Query Language	5	6	8	7
9	Supporting Data Types	6	6	7	8
10	Supporting Mathematical Functions	0	0	7	8
11	Data decomposition	0	0	8	8
12	Language Binding	6	5	7	8
13	Data Storage Limitations	7	5	6	8
Data Model Index =		**71**	**69**	**94**	**98**

TABLE II PWT (Parameter Weightage Table) for DBMS Models

B. *Defining Anchors to the Database Models*

Here we have tried to design and enumerate anchors for respective parameters by asking questions to the user which specifies his requirement with the desired gravity. Anchors are defined for performing parameters only because varying weightages are assigned to the performing parameters. There is no need to define anchors for core parameters because they are basic parameters of any database management system and they all have the common weightage assigned by us. We have defined the following TABLE III which represents the scaling factor for respective performing parameters:

Sr. No.	**Performing Parameters**	**Define Anchors to DBMS Models**			
		HDBMS	**NDBMS**	**RDBMS**	**OODBMS**
1	Data Structure	Good	Poor	Excellent	Very good
2	Establishing Relationship	Weakly supported	Supported	Mostly supported	Strongly supported
3	Query Language	Poor	Good	Excellent	Very good

4	Supporting Data Types	Supported	Supported	Mostly supported	Strongly supported
5	Supporting Mathematical Functions	Not applicable	Not applicable	Mostly applicable	Strongly applicable
6	Data decomposition	No	No	Yes	Yes
7	Language Binding	Supported	Weakly supported	Mostly supported	Strongly supported
8	Data Storage Limitations	Average	Low	Medium	High

TABLE III Defining Anchors for Performing Parameters

The logic behind the process of proposing best database model for the given questionnaire is as follow:
1. In the first step the user has to select anchors of performing parameters that he required.
2. After selecting anchors by the user, we have used the mapping technique between selecting anchors of performing parameters and weightage of performing parameters.
3. The following TABLE IV is used for selecting weightage of the preferred anchors of performing parameters.

Sr. No.	Performing Parameters		Considered Database Models			
			HDBMS	**NDBMS**	**RDBMS**	**OODBMS**
1	Data Structure	**Anchors**	Good	Poor	Excellent	Very Good
		Weight	6	5	8	7
2	Establishing Relationship	**Anchors**	Weakly supported	Supported	Mostly supported	Strongly supported
		Weight	5	6	7	8
3	Query Language	**Anchors**	Poor	Good	Excellent	Very good
		Weight	5	6	8	7
4	Supporting Data Types	**Anchors**	Supported	Supported	Mostly supported	Strongly supported
		Weight	6	6	7	8
5	Supporting Mathematical Functions	**Anchors**	Not applicable	Not applicable	Mostly applicable	Strongly applicable
		Weight	0	0	7	8
6	Data Decomposition	**Anchors**	No	No	Yes	Yes
		Weight	0	0	8	8
7	Language Binding	**Anchors**	Supported	Weakly supported	Mostly supported	Strongly supported
		Weight	6	5	7	8
8	Data Storage Limitations	**Anchors**	Average	Low	Medium	High
		Weight	7	5	6	8

TABLE IV Mapping Anchors to Weightage of Performing Parameters

4. Mapping is done on the basis of above TABLE IV. The table represents the value of respective anchor of performing parameters.
5. After selecting the appropriate weight of anchor we have to find the performance index of proposed database model.
6. Finally, we have to find the data model index by adding the performance index of proposed database model with core parameter weightage i.e. 28 point by using the following formula:

***DMI (Data Model Index) Value** = Performance index + Core Parameter Weightage (i.e. 28) + Defining Standard Weight (i.e. 8)*

This is how we calculate the approximate data model index of the database for given questionnaire.

II. DEFINING A DASHBOARD FOR THE DBMS MODEL INDEX

Here we have defined a dashboard for given DBMS model index. As shown in TABLE II the data model indexes for considered database model is different and it is based on the weightage assigned to each and every parameter of the model. These indexes fall into the following categories:

Sr. No.	Database Models	Data Model Index
1	HDBMS (Hierarchical Database Management System)	**71**
2	NDBMS (Network Database Management System)	**69**
3	RDBMS (Relational Database Management System)	**94**
4	OODBMS (Object Oriented Database Management System)	**98**

We have put it down these data model index on the defining dashboard. Following fig. 3 shows the dashboard for considered DBMS models:

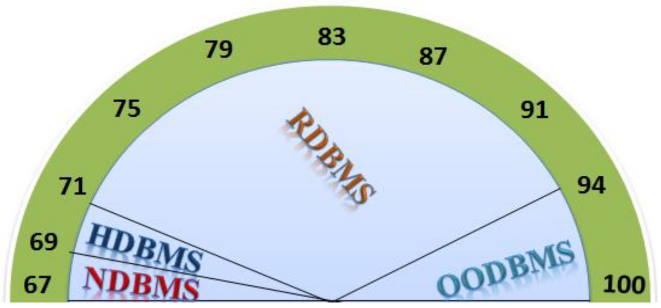

Fig. 3 Dashboard for DBMS Models

The dashboard is used for deciding the value of approximate data model index fall in which category and which database model is suitable for user requirements for storage of questionnaire data.
The given data model index (DMI) Value fall into minimum weightage 67 to maximum weightage 100. Therefore database model index are divided into following categories:

Sr. No.	Database Models	Category
1	HDBMS	70 to 71
2	NDBMS	67 to 69
3	RDBMS	72 to 94
4	OODBMS	95 to 100

The above category is divided in between the minimum and maximum weightage of the database index and it will be vary from minimum value of the data model index i.e. 67 to the next data model index value i.e. 69. Then after, it will vary from 70 to the next data model index value i.e. 71 and so on. The last database model OODBMS lie in between 95 to 100 data model index value.

III. ALGORITHM FOR THE PROPOSED DATABASE MODEL

Here we have defined the generalized algorithm for the proposed database model. The algorithm is used for proposing the better database model to the user as per his requirements. The steps of algorithm are as follows:

Algorithm: To Propose Better Database Model
Inputs: The requirements of the user are satisfied by selecting the defining anchors for the respective parameters.
Outputs: Proposal of the best Database Model that satisfy all requirements of the user.
Step-1: Initialize counter parameters, core parameter and standard parameter.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2015

Initialize Integer variables DMI_Counter = 0, Integer DMI_CoreParameter = 28, Integer DMI_DBStandard = 8
Step-2: Call the functions which return the selected question count and has the return type as integer so that we can verify each database model according to its priority.
1. Integer ooDBMScount = OODBMS_Check()
2. Integer rDBMScount = RDBMS_Check()
3. Integer hDBMScount = HDBMS_Check()
4. Integer nDBMScount = NDBMS_Check()

Step-3: Calculate the total of the values of each selected question and store it in the variable DMI_Counter.
Step-4: Calculate the range of each database model by subtracting static parameter value from the original range. Following are the steps for the new range calculation of each model-
Case-1: The original range defined in our system for OODBMS is **95** to **100**. The fixed parameter is calculated by the formula-
Core Parameter Weightage (i.e. 28) + Defining Standard Weight (i.e. 8) = 36
* Where "Defining Standard" is a parameter
static parameter value is 36 (this is calculated by adding the values of core parameter and defining standard parameter)
The new ranges are calculated by checking the condition mentioned in Step-5 (sub step-1)
Starting range = 95-36 = 59 and
End range = 100-36 = 64
Case-2: The original range defined in our system for RDBMS is **72** to **94**. The fixed parameter are calculated by the formula-
Core Parameter Weightage (i.e. 28) + Defining Standard Weight (i.e. 8) = 36
* Where "Defining Standard" is a parameter
static parameter value is 36 (this is calculated by adding the values of core parameter and defining standard parameter)
The new ranges are calculated by checking the condition mentioned in Step-5 (sub step-2)
Starting range = 72-36 = 36 and
End range = 94-36 = 58
Case-3: The original range defined in our system for HDBMS is **70** to **71**. The fixed parameter are calculated by the formula-
Core Parameter Weightage (i.e. 28) + Defining Standard Weight (i.e. 8) = 36
* Where "Defining Standard" is a parameter
static parameter value is 36 (this is calculated by adding the values of core parameter and defining standard parameter)
The new ranges are calculated by checking the condition mentioned in Step-5 (sub step-3)
Starting range = 70-36 = 34 and
End range = 71-36 = 35
Case-4: The original range defined in our system for NDBMS is **67** to **69**. The fixed parameter are calculated by the formula-
Core Parameter Weightage (i.e. 28) + Defining Standard Weight (i.e. 8) = 36
* Where "Defining Standard" is a parameter
static parameter value is 36 (this is calculated by adding the values of core parameter and defining standard parameter)
The new ranges are calculated by checking the condition mentioned in Step-5 (sub step-4)
Starting range = 67-36 = 31
End range = 69-36 = 33
Step-5: Now we will be checking conditions according to priority of each model using all new ranges which has been calculated within **Step-4**
Priorities of models: OODBMS > RDBMS > HDBMS > NDBMS
1. **If** ooDBMScount > 0 **then** we are supposed to present **OODBMS** model but need to set its Original range after checking with new range

Following steps are written for if the DMI_Counter value falls beyond the original range then it would set it in the original values of given database model

(If DMI_Counter value is less than new range then calculate sub1 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by adding the value of sub1 to the DMI_Counter Else if DMI_Counter value is greater than new range then calculate add1 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by subtracting the value of add1 from DMI_Counter)

```
if  DMI_Counter < 59 then
                Integer sub1 = 59 - DMI_Counter
        DMI_Counter = DMI_Counter + sub1
                Else if  DMI_Counter > 64 then
                Integer add1 = DMI_Counter - 64
        DMI_Counter = DMI_Counter - add1
End if
goto Step-6
```

2. **Else If** rDBMScount> 0 **then** we are supposed to present **RDBMS** model but need to set its Original range after checking with new range

Following steps are written for if the DMI_Counter value falls beyond the original range then it would set it in the original values of given database model

(If DMI_Counter value is less than new range then calculate sub2 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by adding the value of sub2 to the DMI_Counter Else if DMI_Counter value is greater than new range then calculate add2 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by subtracting the value of add2 from DMI_Counter)

```
if  DMI_Counter < 36 then
                Integer sub2 = 36 - DMI_Counter
        DMI_Counter = DMI_Counter + sub2
                Else if  DMI_Counter > 58 then
                Integer add2 = DMI_Counter - 58
        DMI_Counter = DMI_Counter – add2
End if
goto Step-6
```

3. **Else If** hDBMScount> 0 **then** we are supposed to present **HDBMS** model but need to set its Original range after checking with new range

Following steps are written for if the DMI_Counter value falls beyond the original range then it would set it in the original values of given database model

(If DMI_Counter value is less than new range then calculate sub3 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by adding the value of sub3 to the DMI_Counter Else if DMI_Counter value is greater than new range then calculate add3 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by subtracting the value of add3 from DMI_Counter)

```
If  DMI_Counter < 34 then
        Integer sub3 = 34 - DMI_Counter
        DMI_Counter = DMI_Counter + sub3
                Else if  DMI_Counter > 35 then
                Integer add3 = DMI_Counter - 35
        DMI_Counter = DMI_Counter – add3
End if
goto Step-6
```

4. **Else If** nDBMScount > 0 **then** we are supposed to present **NDBMS** model but need to set its Original range after checking with new range

Following steps are written for if the DMI_Counter value falls beyond the original range then it would set it in the original values of given database model

(If DMI_Counter value is less than new range then calculate sub4 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by adding the value of sub4 to the DMI_Counter Else if DMI_Counter value is greater than new range then calculate add4 by subtracting the value of DMI_Counter from the new range and calculate the original range of DMI_Counter by subtracting the value of add4 from DMI_Counter)

```
If  DMI_Counter < 31 then
```

```
            Integer sub4 = 31 - DMI_Counter
    DMI_Counter = DMI_Counter + sub4
            Else if  DMI_Counter > 33 then
            Integer add4 = DMI_Counter - 33
    DMI_Counter = DMI_Counter – add4
End if
Step-6: Stop
```

**The functions content in the step-2, written number of count as integer of the question which is selected by the user to the main algorithm.

IV. RESULTS

We have followed the above algorithm steps and create one application into .net platform. We have designed ASP.net web pages and C#.net language is used for coding. The simple .net application is basically used for accepting requirements from the user for performing parameters by asking questions to them.
Before going to asks questions home page of application display the information or meaning of all parameters to the user. It is mandatory to read all this information before click on the 'OK' button. The output of the given algorithm is to propose a better database model.

Fig 1 describes the number of parameters are used in comparative study of the database models. In the fig 2 shows that how many weightage assigned to each parameter type and fig 3 shows the dashboard of DBMS models. In the TABLE I assigned the maximum weight to each of the considered 13 parameters. TABLE II shows that the PWT (parameter weightage table) for DBMS Models. We have assignedmaximum weight for each DBMS model. TABLE III used for defining anchors for performing parameters and the TABLE IV used for mapping anchors to performing parameters.

V. CONCLUSION

From the above discussion we have come to the conclusion that we have created a new algorithm for proposing a better database model. This algorithm is used by the user for deciding the best data model to his requirement. Through algorithm we asked some questions to the user and proposed them which data model is best suitable as per his requirement. For the implementation proposes we have executed this algorithm in .net platform. We have created web pages in ASP.net and coding is written in C#.net language.

REFERENCES

1. Raghu Ramakrishnan/Johnnes Gehrke, "DatabaseManagement Systems", Second Edition, McGraw-Hill Higher Education.
2. Abraham Silberschatz, Henry Korth, S.Sudarshan, Database System Concepts, 5thEdition, Chapter 1, Introduction, Page 1.
3. Ramez Elmasri, Shamkant B. Navathe, "Fundamentals of Database Systems", 6th Edition, Pearson Education.

BIOGRAPHY

Dr. Ranjana Ingolikar, received Ph.D. degree from RTMNU, Nagpur. She is working as H.O.D. of Department of Computer Science in St. Francis De Sales College, Nagpur.

Mrs. Rasika V. Khandal, received MCA degree from RTMNU, Nagpur in 2008. Since 2009 she is working as Asst. Lecturer in SRPCE, Nagpur in MCA Department.

Dr. Rahul Mohare, received Ph.D. degree from RTMNU, Nagpur. He is working as Asst. Lecturer in DMIMS, Nagpur in MBA Department.