Data Types Diversity Quantitative Indicators in Relational Databases Physical Models

Alexander Aleksandrovich Rybanov, Makushkina Lidiya Aleksandrovna

Volzhskii Polytechnic Institute, Branch of the Volgograd State Technical University, 42a Engels street, 404121, Volzhskii, Russia

Abstract **– The article compares diversity indices applicability used in ecology to assess data types distribution for physical database schemes. The following dominance measures are used to assess the data type's diversity in the physical database scheme: Simpson's diversity index, Shannon diversity index, Simpson's evenness index, Pielu evenness index. A comparative analysis of the data type's diversity indicators for physical database schemes showed that the values of some indices obey certain rules. It is proposed to use diversity indices for physical database schemes' qualitative assessment. The indices considered in this work can also be used to build new models for assessing database complexity.**

Keywords **– diversity index, database, types of data, physical scheme, quality assessment**

1. Introduction

The main influence on the database (DB) operational characteristics is provided by the data physical organization. In order for the database to function correctly and be optimized as much as possible, it is important to carefully select the data types for each field in the tables [1].

DOI: 10.18421/TEM93-04
https://doi.org/10.18421/TEM93-04

Corresponding author: Alexander Aleksandrovich Rybanov, *Volzhskii Polytechnic Institute, Branch of the Volgograd State Technical University, Volzhskii, Russia*
Email: a.a.rybanov@gmail.com

Received: 16 November 2019.
Revised: 18 June 2020.
Accepted: 22 June 2020.
Published: 28 August 2020.

Database developers should know and understand how to use each of them correctly. User-defined data types can significantly strengthen control over the data and increase their integrity [2]. Properly selected field types can reduce the table row's physical size and accordingly the database, making reading and writing procedures faster and more efficient.

2. Recommendations for Field Types Adjusting

Recommendations for the fields adjusting will be examined using MySQL DBMS as an example. MySQL supports a wide variety of data types, and choosing the data storage's right type is critical to good performance. For some fields of the database table data types can be changed without significantly affecting the existing database scheme. Consider the following examples.

The person's age. To store a person's age, it is recommended to use a field with the TINYINT UNSIGNED type, which can store a maximum value of 256. However, someone can use the INTEGER UNSIGNED type without hesitation. But when using the TINYINT data type, 1 byte is required to store the age, in exchange for 4 bytes when using the INTEGER data type.

Types DATE and TIME. DATE and TIME field types occupy 3 bytes each, while a DATETIME field type occupies 8 bytes, but storing all information in one field is more preferable to optimize data processing.

Replacing BIGINT with INT. You can reduce the size of the primary key from 8 to 4 bytes by changing its BIGINT UNSIGNED AUTO_INCREMENT data type to the INT UNSIGNED AUTO_INCREMENT data type. This conversion is possible not only for the primary key but also for all foreign keys that are defined as BIGINT. This approach can significantly reduce the space required for indexes in a highly normalized database.

IP addresses. As a rule, the VARCHAR type (15) is assigned to the database field for storing IPv4 addresses, whose size is on average 12 bytes. In order to reduce the column initial size, it is recommended to define the INT UNSIGNED data type, which requires 4 bytes for the field containing the IPv4 address. IP address translation is possible

using the functions INET_ATON () and INET_NTOA ().

Example:

```
SET @ip = '255.139.67.15';
SELECT @ip, INET_ATON(@ip) AS str_to_i, INET_NTOA (INET_ATON (@ip)) AS i_to_str;
```

MD5 values. A common practice is to store the MD5 value in a field of type CHAR (32). To store the MD5 hexadecimal value more efficiently you can use the UNHEX () and HEX () functions, storing data in the BINARY data type (16). Performing this conversion by half reduces the size of the field.

Example:

```
SET @str = 'wwwvolpiru';
SELECT MD5(@str), LENGTH(MD5(@str)) AS len_md5, LENGTH(UNHEX (MD5 (@str))) AS len_unhex;
```

The PROCEDURE ANALYSE () function allows you to get MySQL database server recommendations on adjusting field types for a table filled with real data, the presence of which plays an important role in decision making. The optimal data types offered by the PROCEDURE ANALYSE () function for each field are focused on solving the table size reducing problem and as a result the entire database.

The syntax for the function using:

SELECT * FROM <*table name*> PROCEDURE ANALYSE([<*max_elements*>, [<*max_memory*]]);

where <*table name*> – is the database table name being analyzed; <*max_elements*> – the maximum different values number contained in the field that the PROCEDURE ANALYSE function uses to check whether the type is ENUM optimal for the field (256 by default); <*max_memory*> – the maximum memory amount that the ENUM field type can occupy (8192 bytes by default).

To exclude ENUM type recommendations in the PROCEDURE ANALYSE function results for fields that contain more than 16 unique values or occupy more than 256 bytes, the following argument values must be specified:

SELECT * FROM <*table name*> PROCEDURE ANALYSE(16, 256);

Keep in mind that the results provided by the PROCEDURE ANALYSE function are just recommendations, the reliability of which is significantly affected by the number of rows in the table. And if in the future the database table is replenished with new data then these recommendations may turn out to be incorrect, therefore the decision on their application remains with the database administrator.

General provisions on the field types application can be summarized in the form of the following recommendations:

- it is recommended to declare fields as NOT NULL;
- it is recommended to use field types that take up less memory;
- for digital data it is not recommended to use character fields;
- it is recommended to use fixed-length fields, as they are processed faster;
- whenever possible, it is recommended to use the types ENUM and SET;
- it is recommended to use specialized types for storing dates and times;
- it is recommended to use the PROCEDURE ANALYSE function to check the characteristics of the table fields filled with real data;
- it is recommended to optimize tables' subject to frequent modification.

3. Problem Statement

The relational database physical scheme contains all the details necessary for a particular DBMS to create the database [7], [9]: tables and columns, names, fields types, primary and foreign keys definitions, indexes.

Physical database scheme quality control is impossible without numerical indicators [4], [8]. In the absence of quantitative measurements, it is difficult to make any design decisions. Existing models for quantifying the relational databases physical schemes complexity [3], [5], [6] do not take into account the data types variety used in them, therefore the task of creating a quantitative criteria system for assessing the data type's diversity used in a physical database scheme is relevant.

Consider the environmental diversity indicators used to solve the problem of assessing the data type's diversity in the physical database scheme for MySQL DBMS.

4. Data Type Diversity Metrics

Diversity is a concept that relates to the variability or difference magnitude between some sets or groups of objects. When evaluating the data type's diversity in a physical database scheme, the following two factors are taken into account:

- species' richness, that is the data types' number included in the physical database scheme;
- the evenness or uniformity of the data types abundance distribution in the physical database scheme.

To assess the data type's diversity in the physical database scheme, we will use the following dominance measures that take into account evenness: the Simpson's diversity index, Shannon diversity index, Simpson evenness index, Pielu evenness index.

The Simpson Diversity Index (D) is calculated using the formula:

$$D = \frac{1}{\sum_{i=1}^{S} p_i^2}, \quad (1)$$

where S – is the data types' number in the physical database scheme (species richness);

p_i – i-th data type fraction in the fields of all types.

The more the Simpson's diversity index approaches the species' richness, the more diverse (in terms of the data types used) is the physical database scheme considered.

The tables 1-2 show the data for calculating the Simpson's diversity index for the MySQL training databases "catalog" and "northwind" physical schemes (**dev.mysql.com**).

Table 1. Data for calculating the Simpson diversity index for the "catalog" database

№	Data type	p_i	The fields of this type number
1	BLOB	0,025974	2
2	DATE	0,012987	1
3	INT	0,519481	40
4	SMALLINT	0,025974	2
5	TEXT	0,116883	9
6	TINYINT	0,012987	1
7	VARCHAR	0,272727	21
8	YEAR	0,012987	1
Species richness S			77

Table 2. Data for calculating the Simpson Diversity Index for the "northwind" database

№	Data type	p_i	The fields of this type number
1	BLOB	0.022472	2
2	DATE	0.056180	5
3	DECIMAL	0.033708	3
4	DOUBLE	0.011236	1
5	INT	0.033708	3
6	MEDIUMINT	0.044944	4
7	SMALLINT	0.089888	8
8	TEXT	0.044944	4
9	TINYINT	0.078652	7
10	VARCHAR	0.584270	52
Species richness S			89

A data type is considered to be dominant in the physical database scheme if its amount is 50% or more of the considered total number types, rare - if less than 10%, and unique - less than 2% (Figure 1.).

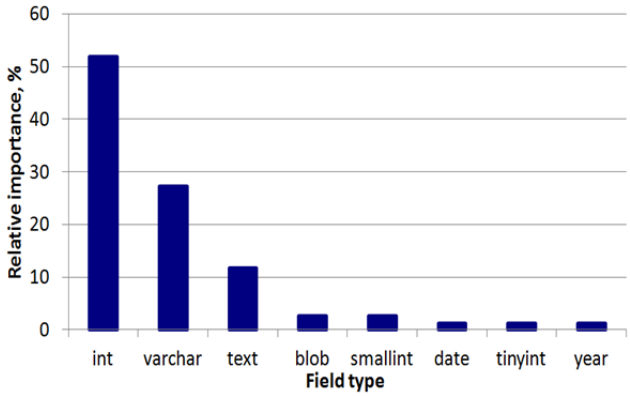

Figure 1. The data types significance curve for the database "catalog" physical scheme

Shannon diversity index (H) is calculated by the formula:

$$H = -\sum_{i=1}^{S} p_i \ln p_i \,. \quad (2)$$

The higher the Shannon index value, the higher the data type's diversity in the physical database scheme.

Simpson's evenness index is calculated by the formula:

$$E = \frac{D}{S}. \quad (3)$$

The closer E approaches to one, the more uniformly the data types are represented in the physical database schema.

The Pielu evenness index (e) is calculated based on the Shannon index:

$$e = \frac{H}{\ln S}. \quad (4)$$

The Pielu index characterizes the data types' evenness in the physical database scheme. The value of the Value Pielu index varies from 0 to 1. The more uniformly are represented the data types in the physical database scheme that comprise it, the closer its value is to unity. Table 3. shows the data type's diversity for various physical database schemes.

Table 3. Data type's diversity indicators for physical database schemes

The physical database schema name	Acronym	Database purpose	*S*	*D*	*H*	*E*	*e*
flight	FL	educational	9	1.9692	0.9508	0.4923	0.6858
world	WO	educational	5	3.0316	1.3115	0.6063	0.8149
music	MU	educational	4	2.1304	0.9911	0.5326	0.7149
employees	EM	educational	5	3.2799	1.3478	0.6559	0.8374
university	UN	educational	4	2.0864	0.9693	0.5216	0.6992
classicmodels	CL	industrial	8	2.5652	1.3559	0.3207	0.652
retailer	RE	educational	8	2.5728	1.3659	0.3216	0.6568
chinook	CH	educational	4	2.3406	0.9907	0.5851	0.7146
contracts	CO	educational	6	3.0422	1.3281	0.5070	0.7413
northwind	NO	educational	10	2.7342	1.5354	0.2734	0.6668
sakila	SA	educational	14	5.4359	1.9983	0.3883	0.7572
catalog	CA	educational	8	2.7797	1.3044	0.3475	0.6273
moodle	MO	industrial	14	3.0758	1.5446	0.2197	0.5853
kadr_oop	KO	industrial	10	2.6165	1.3174	0.2617	0.5722

A data type's diversity indicators' comparative analysis for physical database schemes (Table 3.) shows that training databases are not distinguished from industrial databases by the indices' values, and the indices' values are subject to certain rules.

So, the value of the Simpson evenness index for the considered physical database schemes of varying complexity (Figure 2.) is in the range from 0.2197 to 0.65597. The value of the Pielu evenness index is in the range from 0.57215 to 0.83742.

5. Database Similarity Metrics on the Field Types oCmposition

To solve the two databases' similarity assessing problem by composition in data types in physical schemes, we apply the Jaccard similarity index and Sörensen-Czekanowski similarity index.

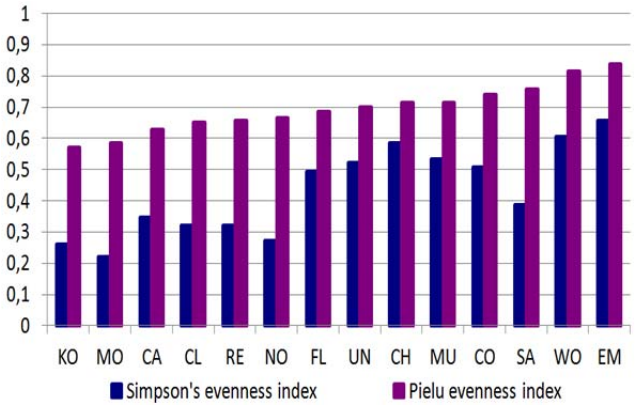

Figure 2. Simpson and Pielu evenness indices

To calculate the similarity in quality terms, we use the Jaccard species composition similarity index which is calculated as:

$$I_J = \frac{a}{a+b+c}, \quad (5)$$

where a – is the common data type number for two databases, b– is the number of types available only in the first database, c – is the number of types available only in the second database.

To assess the quantitative characteristics' similarity we use the and Sörensen-Czekanowski index calculated as:

$$I_s = \frac{2\sum \min(x_{ai}; x_{bi})}{\sum x_{ai} + \sum x_{bi}}, \quad (6)$$

where x_{ai} and x_{bi} – the number of fields of the i-th type in the first and second database; $\min(x_{ai}; x_{bi})$ – the smallest of the i-th type abundance values in the compared databases.

According to the Jaccard index, the field type composition for "catalog" and "northwind" databases is similar to 64%:

$$I_J = \frac{7}{7+1+3} = 0.64.$$

According to the Sörensen-Czekanowski index, the field types composition for the "catalog" and "northwind" databases is similar by 41%, therefore, in this case we may speak about a low similarity degree between the two compared databases:

$$I_S = \frac{2 \cdot (2+1+0+0+3+0+2+4+1+21+0)}{(77+89)} = 0.41.$$

Table 4. shows similarity indices for physical database schemes of varying complexity.

Table 4. Jaccard and Sörensen-Czekanowski similarity indices for physical database schemes

Database acronym	FL	WO	MU	EM	UN	CL	RE	CH	CO	NO	SA	CA	MO	KO
FL	-	0.29	0.33	0.29	0.60	0.20	0.20	0.00	0.11	0.06	0.13	0.04	0.06	0.08
WO	0.60	-	0.29	0.33	0.29	0.18	0.18	0.13	0.22	0.07	0.27	0.10	0.27	0.07
MU	0.27	0.21	-	0.13	0.30	0.05	0.08	0.03	0.02	0.17	0.29	0.04	0.13	0.08
EM	0.26	0.43	0.13	-	0.29	0.30	0.30	0.29	0.57	0.20	0.27	0.24	0.19	0.25
UN	0.62	0.52	0.33	0.18	-	0.20	0.20	0.00	0.11	0.07	0.20	0.08	0.06	0.08
CL	0.08	0.12	0.09	0.36	0.07	-	0.45	0.20	0.27	0.64	0.22	0.51	0.38	0.38
RE	0.08	0.12	0.20	0.36	0.07	0.88	-	0.33	0.56	0.70	0.47	0.56	0.38	0.50
CH	0.00	0.09	0.14	0.25	0.00	0.70	0.75	-	0.43	0.52	0.29	0.64	0.29	0.27
CO	0.05	0.13	0.11	0.39	0.02	0.71	0.78	0.86	-	0.56	0.33	0.61	0.33	0.45
NO	0.17	0.15	0.17	0.25	0.17	0.50	0.80	0.27	0.45	-	0.50	0.41	0.50	0.43
SA	0.03	0.08	0.29	0.10	0.06	0.44	0.51	0.44	0.44	0.63	-	0.33	0.47	0.26
CA	0.20	0.18	0.09	0.30	0.33	0.45	0.78	0.20	0.40	0.64	0.47	-	0.29	0.38
MO	0.00	0.01	0.01	0.01	0.00	0.05	0.05	0.05	0.05	0.09	0.11	0.04	-	0.50
KO	0.01	0.02	0.00	0.09	0.00	0.22	0.24	0.26	0.29	0.30	0.27	0.28	0.14	-

The Table 4. upper part shows the Jaccard similarity coefficients values (I_J), and the lower part shows the Sörensen-Czekanowski similarity coefficients values (I_S). For example, from the data in Table 4., it follows that when assessing the similarity of the RE, CH, CO database projects with the MO database project, the indices' values I_S taking into account qualitative features are equal to:

$$I_S(RE,MO) = I_S(CH,MO) = I_S(CO,MO) = 0.05.$$

At the same time, when assessing the similarity of the RE, CH, CO database projects with the MO database project, the indices values I_J, that take into account quantitative characteristics, have completely different meanings:

$$I_J(RE,MO) = 0.38,\ I_J(CH,MO) = 0.29,$$
$$I_J(CO,MO) = 0.33.$$

As a measure that simultaneously takes into account the qualitative and database similarity quantitative signs in the field types' composition we take the following integral index:

$$I = w_1 I_J + w_2 I_S, \qquad (7)$$

where w_1, w_2 are the weights for the Jaccard and Sörensen-Czekanowski indices.

This metric can be considered as an additional indicator in the automated analysis process for the plagiarism presence in the physical database schemes' source codes. As well as dividing the database by similar properties in order to study the database projects quality.

6. Conclusion

The considered data type's diversity *indicators (1) - (4)* in the physical database schemes supplement the currently existing databases metric characteristics. The data type's diversity indicators' values distribution laws, Additional studies on large databases collections, will allow the rules set formation for the physical database schemes' qualitative assessment.

The proposed quantitative indicators' system *(5) - (7)* expands the criteria set for analysis of plagiarism presence in the source program code.

References

[1]. Piattini, M., Calero, C., & Genero, M. (2001). Table oriented metrics for relational databases. *Software Quality Journal*, *9*(2), 79-97.

[2]. Pavlic, M., Kaluza, M. & Vrcek, N. (2002). Database complexity measuring method. *Proceedings of the ISRM 2002 Conference*, Las. Vegas, NV, USA.

[3]. Rybanov, A.A., Sviridova, O.V., Korotkova, N.N., Lyasin, D.N. & Abramova, O.F. (2019). Model for estimating the complexity of physical schema for relational database. *Inženernyj vestnik Dona, 3*(54).

[4]. Calero, C., Piattini, M., & Genero, M. (2001, June). A case study with relational database metrics. In *Proceedings ACS/IEEE International Conference on Computer Systems and Applications* (pp. 485-487). IEEE. DOI: 10.1109/AICCSA.2001.934049

[5]. Paul, R. A., Kunii, T. L., Shinagawa, Y., & Khan, M. F. (1999). Software metrics knowledge and databases for project management. *IEEE Transactions on Knowledge and Data Engineering*, *11*(1), 255-264. DOI: 10.1109/69.755633

[6]. Calero, C., Piattini, M., & Genero, M. (2001). Metrics for controlling database complexity. In *Developing quality complex database systems: practices, techniques and technologies* (pp. 48-68). IGI Global. DOI: 10.4018/978-1-878289-88-9.ch003

[7]. Subali, M. A. P., & Rochimah, S. (2018, July). A new model for measuring the complexity of SQL commands. In *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)* (pp. 1-5). IEEE. DOI: 10.1109/ICITEED.2018.8534782

[8]. Justus, S., & Iyakutti, K. (2007, October). Assessing the Object-level behavioral complexity in Object Relational Databases. In *IEEE International Conference on Software-Science, Technology & Engineering (SwSTE'07)* (pp. 48-56). IEEE. DOI: 10.1109/SwSTE.2007.12.

[9]. Lin, X. Y. (2011). A Relational Database Complexity Computing Model. In *Key Engineering Materials* (Vol. 474, pp. 1464-1469). Trans Tech Publications Ltd. DOI: 10.4028/www.scientific.net/KEM.474-476.1464