інструментальними системами, не забезпечують велике число агентів і високу швидкість роботи і т.д. [5]

У зв'язку з цим справжня робота присвячена вирішенню проблеми підвищення ефективності використання ресурсів при оперативному управлінні проектами створення та експлуатації зразків нового програмного забезпечення в організаціях за рахунок впровадження мультиагентних технологій в організації з дивізійною (холістичною) структурою.

В якості випадку застосування досліджуваного підходу обрана компанія, велике значення для якої має взаємодія та ефективність співробітників, відділів співробітників, а також ефективне планування та виконання проектних завдань .

Для аналізу даної проблеми обрана компанія - «Template Monstrs», що займається створенням та реалізацією, і є одним з найбільших в світі постачальників шаблонів для сайтів.

Цільовим призначенням роботи є проведення досліджень про впровадження мультиагентних технологій в організації з дивізійною (холістичною) структурою.

Наукова новизна роботи полягає у застосуванні методів підтримки організації на основі мультиагентного підходу.

Перелік джерел посилання.

1. Виттих В.А, Скобелев П.О. Мультиагентные модели взаимодействия для построения сетей потребностей и возможностей в открытых системах // Автоматика и телемеханика. - 2013. -№1,-С. 177-185.

2. Скобелев П.О. Холистический подход к созданию открытых мультиагентных систем // Тр. III Междунар. конф. по пробл. упр. и моделирование. в сложных системах, Самара, 4-9 сентября, 2001. - Самара- СНЦ РАН, 2010. - С. 147 - 160.

3. Андреев В.В., Минаков И.А., Пшеничников В.В., Симонова Е.В., Скобелев П.О. Основы построения мультиагентных систем. – Самара: изд-во ПГУТИ, 2007. – 290 с.

4. Абрамов Д.В., Андреев В.В., Симонова Е.В., Скобелев П.О. Открытые мультиагентные системы для принятия решений в задачах динамического распределения ресурсов. – Самара: изд-во ПГУТИ, 2008. – 290 с.

5. George Rzevski. FAQ On Agents and Multi-Agent Systems // http://www.naun.org/journals/educationinformation/eit-11.pdf 5. European Coordination Action for Agent-based Computing // http://www.agentlink.org

UDC 004.65

Orlovskyi D.L., *Ph.D., Associate Professor of the Department of Software Engineering and Management Information Technologies*

Kopp A.M., *Ph.D., Associate Professor of the Department of Software Engineering and Management Information Technologies*

AN APPROACH TO THE TRANSLATION OF NATURAL LANGUAGE BUSINESS RULES INTO SQL TABLE CONSTRAINTS

National Technical University «Kharkiv Polytechnic Institute», Ukraine

Introduction. Modern software applications, from mobile applications for Android or iPhone smartphones to enterprise-level information systems and e-commerce websites, cannot be developed without components for persistent data storage. Databases and database management systems that appeared in the late 70s of the previous century were supposed to solve the problem of persistent and independent data storage for applied software solutions. Database management systems (DBMS) are specialized software systems that manage database structures, make collections of data persistent and shareable in a secure way [1].

According to the latest rating of the most popular DBMS provided by DB-Engines Ranging for November 2021 [2], the top four of five in total database management systems are relational table-oriented DBMS:

– Oracle – relational DBMS with multi-model capabilities, such as document store, graph DBMS, RDF (Resource Description Framework) store, and spatial DBMS;

– MySQL – relational DBMS with multi-model capabilities, such as document store and spatial DBMS;

– Microsoft SQL Server – relational DBMS with multi-model capabilities, such as document store, graph DBMS, and spatial DBMS;

– PostgreSQL – relational DBMS with multi-model capabilities, such as document store and spatial DBMS.

All of these database management systems require a certain level of training for efficient database design, development, and maintenance when used in production enterprise projects. First of all, such DBMS require proficient SQL (Structured Query Language) language skills, the experience of working with real-world databases, and an understanding of SQL dialects supported by different database management systems (e.g. PL/SQL in Oracle, T-SQL in Microsoft SQL Server PL/pgSQL in PostgreSQL etc.).

Problem statement. Hence, it could be a challenge for inexperienced software engineers to design, extend, or maintain a production database without making critical errors that will possibly affect all the business. Moreover, the modern software engineering industry considers Agile software development processes [3] with the same team members responsible for database design, who are responsible for overall backend programming. Thus, lack of skills and, sometimes, the experience could lead to poor design solutions or even mistakes of database development.

Related work. Existing studies in the field of business rules translation into SQL database scripts [4], [5] have considered the most translation of text queries for data retrieval into SELECT SQL statements. Inspired by the presence of unsolved yet problems, we have proposed an idea of database generation SQL scripts retrieval from natural language business rules provided as facts about a certain domain [6].

Later we have elaborated our idea and published the full paper, where the method, algorithms, and software prototype were proposed [7]. However, in [6] and [7] we addressed only so-called "facts" business rules that could be provided by business analysts after the preliminary study of the subject domain. However, in real databases, the structure is as much important as further data consistency and integrity. Therefore, the approach proposed earlier should be extended with "constraints" support according to Wiegers classification (see Figure 1) [8].

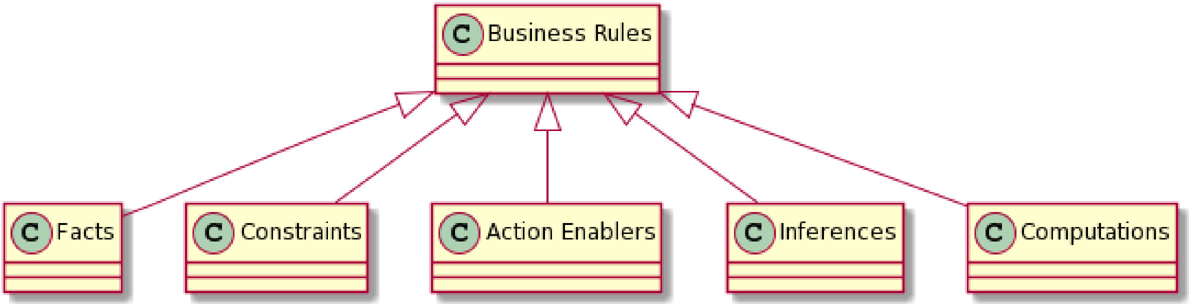


Fig. 1. Business rules classification according to Wiegers [8]

Research aims. This paper aims to the development of an approach to translate natural (or almost) language business rules into SQL scripts for database tables generation with consideration of data integrity and consistency constraints. Therefore, the following tasks should be solved in this study in order to reach the aim:

– propose an approach to the translation of natural language business rules into SQL table constraints;

– verify proposed approach by performing necessary calculations;

– discuss obtained results, summarize the work done, and define future work directions.

Proposed approach. The proposed approach is based on regular expressions, also used before in our studies [6] and [7], where we have described algorithms to generate DDL (Data Definition Language) from "facts" business rules.

However, in this short paper, we consider only essential constraints of the MySQL database management system, such as the following with corresponding business rule patterns:

– CHECK, check_rule ::= ⟨column⟩ {of} ⟨table⟩ {should be} ⟨operator⟩ ⟨value⟩;

– DEFAULT, default_rule ::= {by default} ⟨column⟩ {of} ⟨table⟩ {should take} ⟨value⟩;

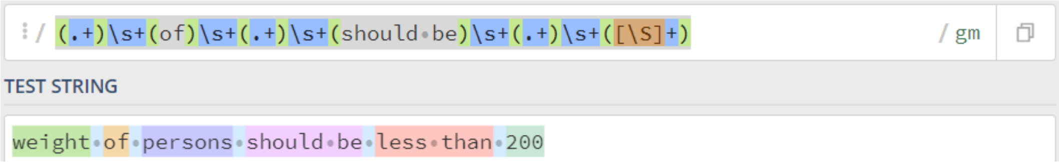
– NOT NULL, not_null_rule ::= ⟨column⟩ {of} ⟨table⟩ {is required}.

The CHECK constraint in MySQL, as well as similar constraints in other relational DBMS, should contain conditions built using comparison operators, such as equal (“=”), greater than (“>”), less than (“<”), greater than or equal (“>=”), less than or equal (“<=”), and not equal (“<>” or “!=”).

Therefore, we may introduce an additional rule for the operators mentioned in business rules and which then will be mapped to mentioned arithmetical operators: operator ::= ⟨greater than| less than|greater than or equal to|less than or equal to|equal to|not equal to⟩. Otherwise, if any other operators are given, the business rules should not be processed.

To process the expected structure of natural language business rules, we have come up with regular expressions to retrieve all of the considered MySQL constraints (mentioned above).

Results. The regular expression for CHECK MySQL and results that demonstrate its testing are shown below in Figure 2.



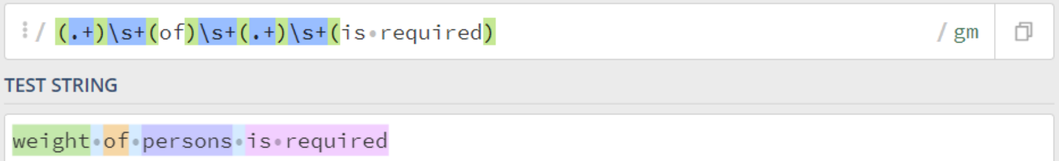
```
/ (.+)\s+(of)\s+(.+)\s+(should be)\s+(.+)\s+([\S]+) / gm
```

TEST STRING

```
weight of persons should be less than 200
```

Fig. 2. Regular expression testing for CHECK constraints

The regular expression for DEFAULT MySQL and results that demonstrate its testing are shown below in Figure 3.



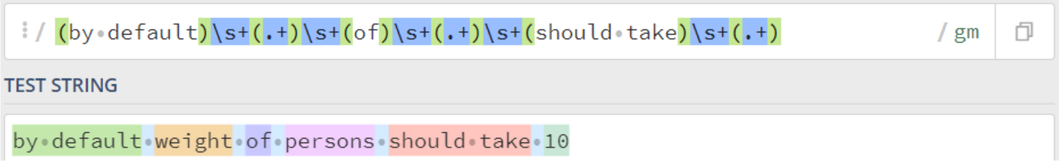
```
/ (.+)\s+(of)\s+(.+)\s+(is required) / gm
```

TEST STRING

```
weight of persons is required
```

Fig. 3. Regular expression testing for DEFAULT constraints

The regular expression for NOT NULL MySQL and results that demonstrate its testing are shown below in Figure 4.



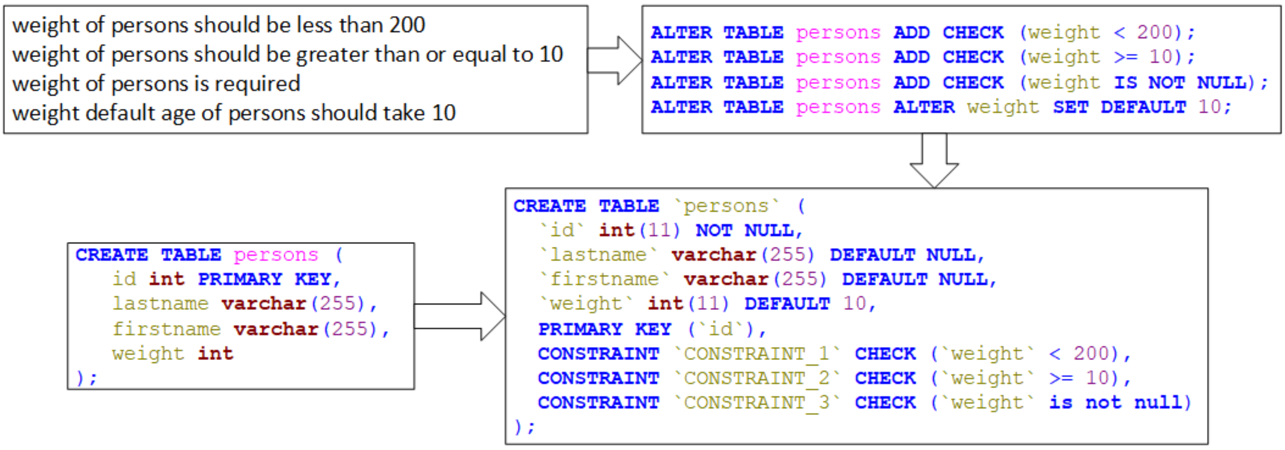
```
/ (by default)\s+(.+)\s+(of)\s+(.+)\s+(should take)\s+(.+) / gm
```

TEST STRING

```
by default weight of persons should take 10
```

Fig. 4. Regular expression testing for NOT NULL constraints

Hence, using the regular expressions demonstrated above, these sample business rules could be translated into the MySQL statements in order to provide necessary constraints (see Figure 5).



weight of persons should be less than 200
weight of persons should be greater than or equal to 10
weight of persons is required
weight default age of persons should take 10

```
ALTER TABLE persons ADD CHECK (weight < 200);
ALTER TABLE persons ADD CHECK (weight >= 10);
ALTER TABLE persons ADD CHECK (weight IS NOT NULL);
ALTER TABLE persons ALTER weight SET DEFAULT 10;
```

```
CREATE TABLE persons (
   id int PRIMARY KEY,
   lastname varchar(255),
   firstname varchar(255),
   weight int
);
```

```
CREATE TABLE `persons` (
  `id` int(11) NOT NULL,
  `lastname` varchar(255) DEFAULT NULL,
  `firstname` varchar(255) DEFAULT NULL,
  `weight` int(11) DEFAULT 10,
  PRIMARY KEY (`id`),
  CONSTRAINT `CONSTRAINT_1` CHECK (`weight` < 200),
  CONSTRAINT `CONSTRAINT_2` CHECK (`weight` >= 10),
  CONSTRAINT `CONSTRAINT_3` CHECK (`weight` is not null)
);
```

Fig. 5. Regular expression testing for NOT NULL constraints

Conclusion and future work. In this paper, we proposed the approach for translating natural language business rules into SQL scripts for database tables generation with consideration of data integrity and consistency constraints. The approach is based on regular expressions, which were verified for MySQL CHECK, DEFAULT, and NOT NULL constraints on simple examples. In the future, the exact algorithm of constraints business rules translation into SQL statements should be elaborated and tested on real databases, and detected limitations should be eliminated.

References.

1. Coronel C., Morris S. Database systems: design, implementation, & management. – Cengage Learning, 2016.

2. DB-Engines Ranking. [Electronic resource]. Access mode : https://db-engines.com/en/ranking

3. Ahmed A., Prasad B. Foundations of Software Engineering. – CRC Press, 2016.

4. Moschoyiannis S., Marinos A., Krause P. Generating SQL queries from SBVR rules // International Workshop on Rules and Rule Markup Languages for the Semantic Web. – Springer, Berlin, Heidelberg, 2010. – P. 128–143.

5. Kate A. et al. Conversion of natural language query to SQL query // 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA). – IEEE, 2018. – P. 488–491.

6. Kopp A. M., Orlovskyi D. L. Towards the approach to database structure generation from business rules based on natural language expressions // Information technologies and automation – 2020. – Odessa : ONAFT, 2020. – P. 224–226.

7. Kopp A., Orlovskyi D., Orekhov S. An Approach and Software Prototype for Translation of Natural Language Business Rules into Database Structure. [Electronic resource]. Access mode : http://ceur-ws.org/Vol-2870/paper94.pdf

8. Wiegers K., Beatty J. Software requirements. – Pearson Education, 2013.