REPRESENTATION OF COMPLEX OBJECT AND PROCESS STRUCTURES IN GRAPH DATA BASE ON THE EXAMPLE OF AGRICULTURAL TRACTOR

ROBERT PIETRZYK[1], ANDRZEJ MARCINIAK[2]

Abstract

The paper presents an application of Graph Data Base technology in knowledge engineering task concerned with knowledge representation of complex structures. The first example is graph representation of a composition of tractor gearbox. It is an example of application of static partonomy relation. The adequate codding of that relation in Graph Data Base is easily expressed in Cypher language. There were presented some applications of Graph Data Base useful in the management of the technical object existence in different "life" phases, as design, production, maintenance and recycling. The second example concerns the Graph representation of activity structure. Such structures were built based on an agricultural tractor's technical revision system. Created Data Base allows for processing the queries referred to the questions of what activities in what period should be performed to maintain the agricultural tractor ready to work in good technical condition.

The Graph Data Base technology is the first step to create semantic systems for data storing and processing in order to extract knowledge and information useful in precise physical process management.

Keywords: Graph Data Base, Data Engineering, Knowledge Representation, Tractor's BOM (Bill of Material), Technical Revisions System.

1. Introduction

The structure of complex facilities is defined by the term BOM (Bill of Material) [2]. The precise specification of a technical object structure is used in all stages of its existence - from construction through production, operation and maintenance phases to recycling. Data describing the structure is an important element of the material flow logistics. There are many approaches for analytical representation BOM structures [7]. Representation of structure in classical, relational databases is not very effective, because the number

[1] Transport and Computer Science, University of Economics and Innovation in Lublin, Projektowa 4, 20-209, Lublin, Polska, e-mail: robert.pietrzyk@wsei.lublin.pl

[2] Transport and Computer Science, University of Economics and Innovation in Lublin, Projektowa 4, 20-209, Lublin, Polska, e-mail: andrzej.marciniak@wsei.lublin.pl

of parts of a complex object can reach thousands as well as the number of relations between them. Apart from the labour-intensive nature of implementing such a database, it is difficult to update it. Each extension or reduction of the number of elements requires changes in the entire database. The most important advantage of the graph databases is the reduction of search and upgrade operations to the detection and evaluation of paths in the graph [6, 11].

In addition, the semantics of structural relationships represented explicitly by graph arcs enables data-driven machine learning and algorithmic inference.

2. Complex object BOM representation in graph databases GDB

Theoretical basis of graph databases is the theory of graphs, [1, 16, 18]. The graph is a mathematical entity (abstraction) defined as a pair of sets (N, A), where N is a set of nodes and A is a set of relationships between nodes. Mathematical definition of the graph has a well-defined semantics and in terms of logic it corresponds to the predicate's logic. This makes graph language an expressive system of knowledge representation about complex structure objects and processes.

Graph nodes represent objects of the physical world and relationships determine how they constitute a complex system. These relationships may be expressed with predicates such as *is_a*, *has_a* and their extensions. This allows to represent symbolically the knowledge of *what exists* in the subject domain, and what properties it has. Extensions to these relationships, such as *part_of* and *cause/effect_of* are used in various logic inference schemes (induction, deduction, abduction) to describe and reason about an object's structure and processes. Specified relations are important in problems of ontological [9] knowledge representation, particularly that involves meronimic approach [13, 14]. From the pragmatic point of view, the language of graphs makes it possible to construct symbolic, highly modular and recursive representation of knowledge.

In a particular case, the nodes of the graph represent components (parts) of a complex object. A node may have multiple labels that are used to group nodes into sets. Labels are indexed to accelerate finding nodes in the graph. Nodes can have one or more properties (attributes) stored as key/value pairs). Nodes are connected to other nodes via relationships. Relationships are directional, connect two nodes and can have one or more properties (i.e., attributes stored as key/value pairs). Properties are named values where the name (or key) is a string. Properties can be indexed, and constrained, composite indexes can be created from multiple properties, [25].

Properties determine the role of nodes and, in the case of relationships, specify the flow between nodes. The term "flow" is a good metaphor when a process view of complex structures is needed.

In BOM structure, relationships represent predicate *part_of* or *has_a_part*. In a simple case, the graph representing the complex object is a *Tree*, but when a higher-order structural levels have common downstream components, it becomes a *Network*.

Technology of graph databases can be applied not only to the semantic representation of the complex objects structure but also to the structure of processes. Particularly useful is a representation of complex processes comprising all phases of a technical object existence [2]. Technical object existence starts from its conceptualization phase, through design, production, operation and maintenance to recycling. The GDB can be used to record the facts that occur in the process. That register of events is required by standards for tracing and tracking operation, maintenance and repair services [4], emission processes [3], the state of road infrastructure [15], measurements of Loyalty [8], documentation and modelling of all transport's risks [10]. Graph representation of process participants and relationships between them provides not only factual data but also meta data needed to extract from data their semantic content.

BOM as an example of GDB application can be generalized to BOP (Bill Of Process). Then, for example, the graph nodes represent activities/ operations that create different levels of process description and each node can be described with any number of properties, e.g. the purpose of an action, resources needed to perform it, criteria for correctness of its execution, etc. Relationships may be temporal interdependencies, necessary to organize and track the process over time. Each operation then has a specific start time, duration, end time, determined with accuracy to the probability distribution. Due to temporal relations, we know the temporal structure of operations in and the possibility of replacing one operation with several others, e.g. the operation of the whole assembly replacement is substituted by the exchange of specific parts, which may result with different cost and time.

3. GDB Design with NEO4J

A convenient tool for building graph database is Neo4J, [5, 12, 17, 21, 22, 26]. In this system, all operations on the GDB from creation through searching and updating are performed using queries coded in the CYPHER language, [24].

The correct sentence in that graphs language has a form: (node)-[relationship]->(node).

We use the word sentence here, because this structure is exactly the sentence structure: (subject) - [predicate] - (object), used in many natural languages. The graph is the equivalent of sentences in which relations represent predicates of sentences and the nodes of the graph are predicate arguments. In order to put a graph segment: (node) - [relation] -> (node) in the database, it should be preceded by the word CREATE or MERGE. We suggest using the MERGE command, because it does not duplicate the sentences previously introduced in contrast to CREATE. The definition of a node has the following scheme:

(variable: label {property1, property2,..., propertyN })

where property is a couple {Variable: value}. The value of a variable can be a number or a string, and must be enclosed in quotation marks, for example {Color: 'red'}. A record of a relationship has a similar pattern:

[:relationship_name {property1, property2,..., propertyN }]

where properties, as of nodes, they are pairs of {Variable: value} eg. {Distance: '10 km', Flow: '100 car/hour '}. The sentence in the CYPHER language takes the form:

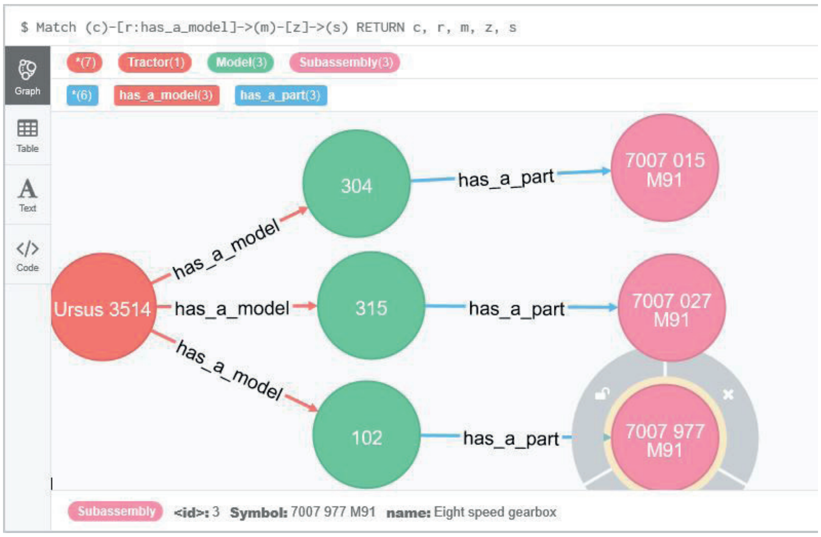
MERGE (variable1: label1 {List of node1 properties) MERGE (variable2: label2 {List of node2 properties}) MERGE [:relationship_name { List of relationship propeties}];

CYPHER language is so expressive as SQL. Specification of CYPHER language is shown in [23, 14].

4. Example of building GDB

As an example, the BOM of the URSUS 3514, models 102, 304, 315 the unit gearbox was selected. The data from the spare parts catalogue [20] were used. The gearboxes in each model differ from each other and for that we construct a graph whose input node is represented by the 3514 tractor and three of its models: 102, 304 and 315. The query Q1 (described precisely in supplement to the paper) creates the subsequent structural levels of the tractor URSUS 3514, produced in three models: 102, 304, 315 built from hundreds components and parts, from which only gearbox will be the subject of next consideration created in query Q2 (see in supplement).

Query Q3 displays graphically (Fig. 1.) the result of queries Q1 and Q2.


```
$ Match (c)-[r:has_a_model]->(m)-[z]->(s) RETURN c, r, m, z, s
```

Subassembly <id>: 3 Symbol: 7007 977 M91 name: Eight speed gearbox

Fig. 1. Result of query Q3 (displaced as a screen-print from Noe4J, showing graph interactives, red colour represents note of URSUS tractor 3514, green colour – models of tractor, pink colour – three types of gearbox, blue colour represents relation *has a part* and red colour *has a model*; properties of the clicked note 7007 977 M91 is listed as: **subassembly** ⟨id⟩ Symbol: **7007 977 M91** name: **Eight speed gearbox**).

Thanks to Neo4J interactives of each graph it is possible to see immediately all properties of the clicked node or relations in between.

Each gearbox consists of many parts, most of them are common in all models of tractors. Some parts appear repeatedly in each model. Cypher query Q4 (see in supplement) adds to GDB parts of gearbox model 102 and corresponding graph as a result of query Q5 (see in supplement) is shown in Fig. 2.

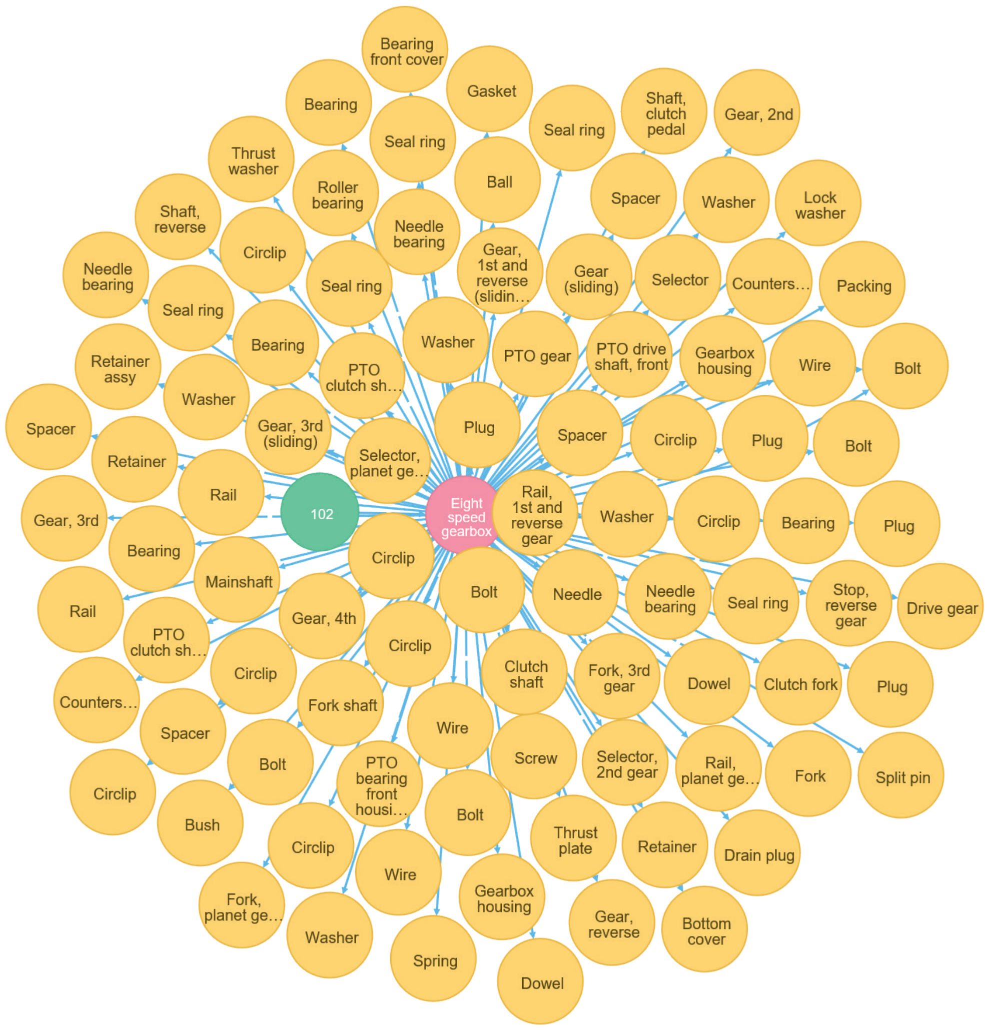


Fig. 2. Graph of gearbox parts, model 102 as a result of query Q5

When all parts are attached into all three models, we get Graph (Fig. 3.), showing the relations between tractor and models, models and gearboxes, gearboxes and parts from which each of gearbox is made.

Having such relations, it is possible to create (ask) many practical questions related to presented Graph Data Base (Figs. 1-3) for example:

1. How many parts are included in the gearbox of URSUS tractor 3514 model 102?
2. What and how many common parts are included in the gearbox of model 102, 304, 315?
3. What identification code belongs to searching part?

The individual parts or components of each gearboxes are presented in Fig. 3. as yellow nodes with running numbers. It is clearly presented, that some part belongs only to one gearbox (for example parts 128, 129, 130, 131 are mounted only in gearbox 7007 027 M…), some parts belong to two gearboxes (for example 108, 115, 117) but most of them belong to all three gearboxes, as common parts.

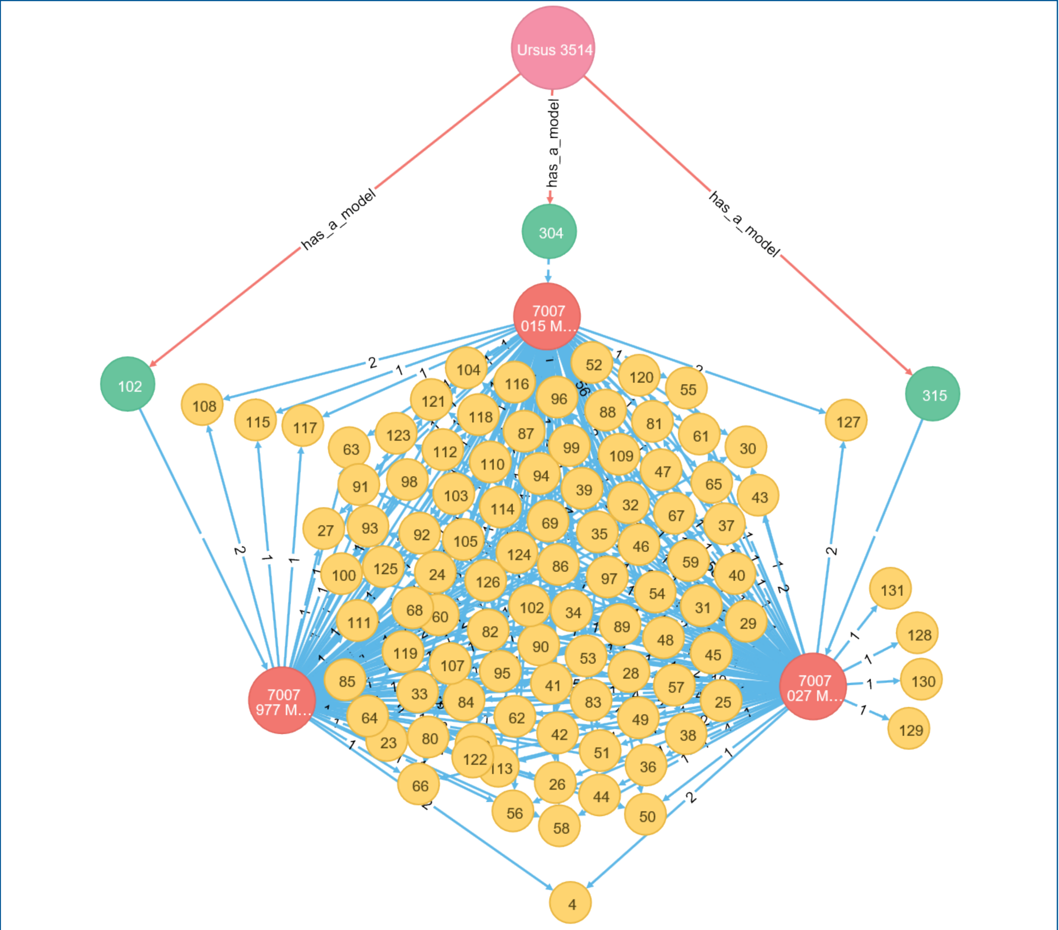
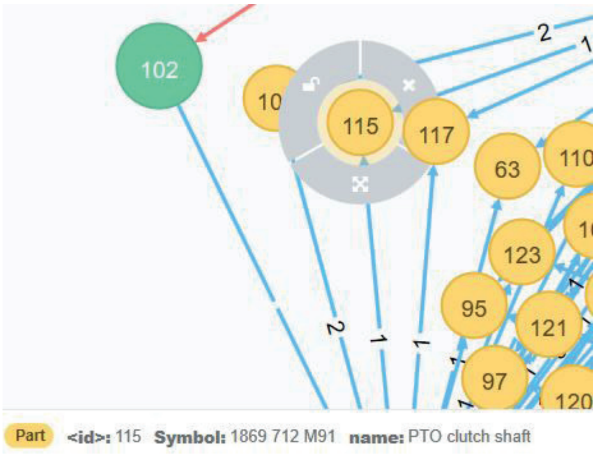


Fig. 3. Graph of gearbox, model 102, 304, 315

Graph in Fig. 3. is too big to display readable names of gearbox parts and components. When the graph is displayed on the screen in Neo4J, then after clicking on chosen node, thanks to graph reactiveness, all descriptions of each node like node name and other properties can be found out. Such possibility is shown in Fig. 4.,



Part <id>: 115 Symbol: 1869 712 M91 name: PTO clutch shaft

Fig. 4. Graph representing additional explanation of Fig. 3., (by clicking the node with running number 115, the identification code: 1869 712 M91(from BOM) is appeared as well as the part name: PTO clutch shaft).

It is possible to create more questions if we enter additional attributes (properties) to the nodes and relations; for example: parts price, parts weight, material that the part is made of, spare parts producer's name, climatic condition in which the parts are properly working, substitute of given parts, the year of tractor's or parts production; then the questions could be:

1. What is the material's price of gearbox assembled to given model or price of chosen components and parts (it is an important topic for logistic specialists who are working on price optimization of final product)?
2. What is the weight of a given model's gearbox or its components?
3. What type of materials are implemented in the gearbox of given model (important question in recycling activity)?
4. What is the name of the certain spare part's producer?
5. Does the pointed part suitable to work in tropic climate condition (important topic for the designers, technologist and foreign traders planning the export of final product to different destinations all over the world)?
6. What substitutes of original parts are allowed to use in certain circumstances of a tractor working (this question can concern the foreign markets)?
7. What spare part is suitable to be installed in the gearbox of the tractor's model made in certain years?

Expanding BOM approach to the whole tractor, the questions presented above will relate to all components, subcomponents, parts and operating liquids.

In analogy to the construction of the graph that represents a BOM, it is possible to create graphical representations of Processes. This is illustrated on the example of technical inspections of the Ursus MF255 tractor, according to data specifying in the repair instructions [19]. According to this manual, there are 5 types of technical revisions P-1 ... P-5. Technical revision includes 8 inspection objects (engine, fuel system and air filter, ..., cabin) and in each of them there are 3 to 6 activities such as check and fill, clean, replace (Table 1.).

Table 1. Technical Revisions of Ursus MF255

Technical revision number	**P-1**	**P-2**	**P-3**	**P-2**	**P-4**	**P-2**	**P-3**	**P-2**	**P-5**
The number of moto hours for making the each technical revision	**10**	**123**	**250**	**375**	**500**	**625**	**750**	**875**	**1000**
ENGINE									
check the oil level in engine's oil sump and fill it up if necessary	1	1	1	1	1	1	1	1	1
change the oil in oil sump			1		1		1		1
change the oil filter			1		1		1		1
check and adjust the clearance of engine valves					1				1
clean the engine ventilation host					1				1
FUEL SYSTEM and AIR FILTER									
check the condition of glass fuel filter settler and drain the water	1	1	1	1	1	1	1	1	1
change the cartridge of fuel filter			1		1		1		1
verify the condition of injectors					1				1
drain and clean the fuel tank									1
check the liquid level in settler and oil level in air filter	1	1	1	1	1	1	1	1	1
clean the cartridge and change the oil in wet air filter			1		1		1		1
COOLING SYSTEM									
check the cooling liquid level and fill it up	1	1	1	1	1	1	1	1	1
clean the radiator ribs			1		1		1		1
drain, clean and fill up the cooling system					1				1
ELECTRIC SYSTEM									
check the electrolyte level in battery and fill it up with distilled water		1	1	1	1	1	1	1	1

Table 1. (cont.)

clean the upper battery surface and grease the clamps with technical Vaseline			1		1		1		1
check the belt tension and adjust properly		1	1	1	1	1	1	1	1
check the battery									1
STEERING SYSTEM									
check the oil level in steering system and fill it up					1				1
check the bearing pressure of the front wheel hubs and adjust			1		1		1		1
check the geometry of the front wheels and adjust					1				1
POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM									
check oil level in gearbox and fill it up		1	1	1	1	1	1	1	1
clean the transmission oil filter					1				1
change the oil in powertrain system and planetary gears					1				1
check the oil level in powertrain system			1		1		1		1
check and adjust the differential lock pedal					1				1
CLUTCH AND BRAKES									
check the clutch pedal idle and adjust		1	1	1	1	1	1	1	1
check the brakes and adjust			1		1		1		1
check the work of pneumatic installation and brake valves			1		1		1		1
CABIN									
check and fill up the window spray liquid	1	1	1	1	1	1	1	1	1
clean the cabin filter	1	1	1	1	1	1	1	1	1
change the cabin filter									1
check and tighten the cabin fastening screws						1			1

Graph database representing technical revisions was created as well in Neo4j system.

Some example queries Q7, Q8, Q9 (description see in supplement) are shown below. Resulting answer drawings are self-explanatory.

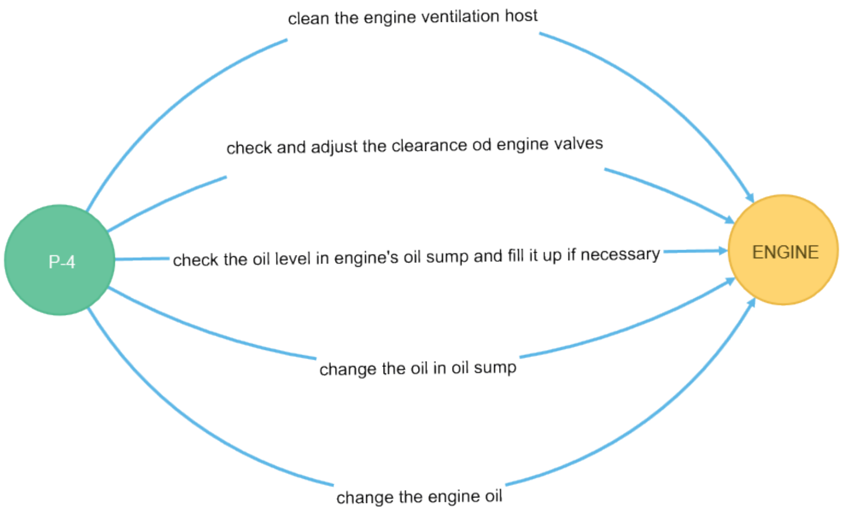

P-4
ENGINE
clean the engine ventilation host
check and adjust the clearance od engine valves
check the oil level in engine's oil sump and fill it up if necessary
change the oil in oil sump
change the engine oil

Fig. 5. Result of query Q7: what actions are performed on ENGINE revision P-4

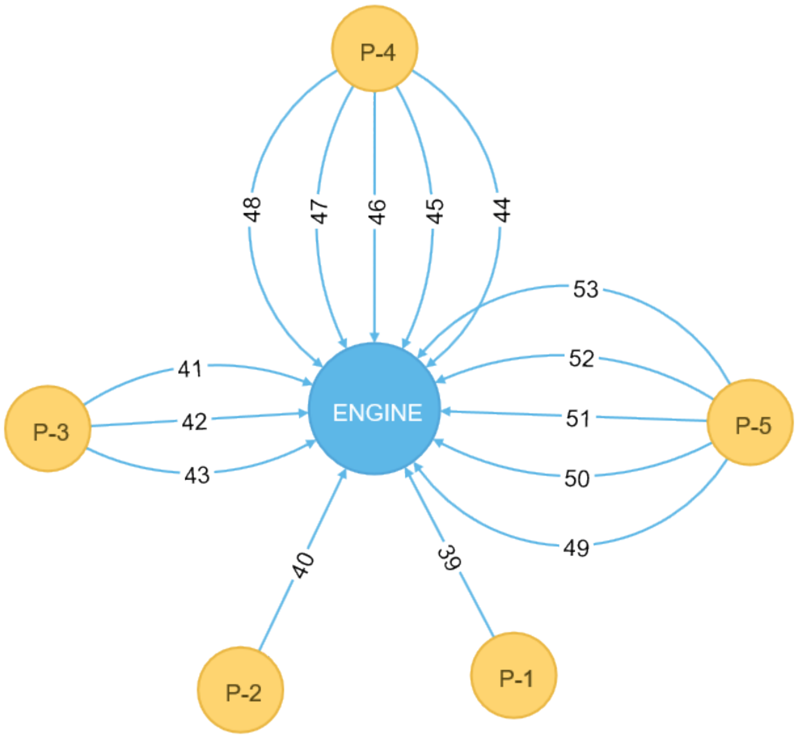


Fig. 6. Result of query Q8: what actions are performed on ENGINE during each revision

The relation in Fig. 6 represent actions of technical revisions operations listed in Table 1. For example: relation 49 means "check the oil level in engine's oil sump and fill it up if necessary". That information is available due to the graph interactive or a query.

The same explanation is valid also to the relations represented on Fig. 7., on which the index points the actions from Table 1.

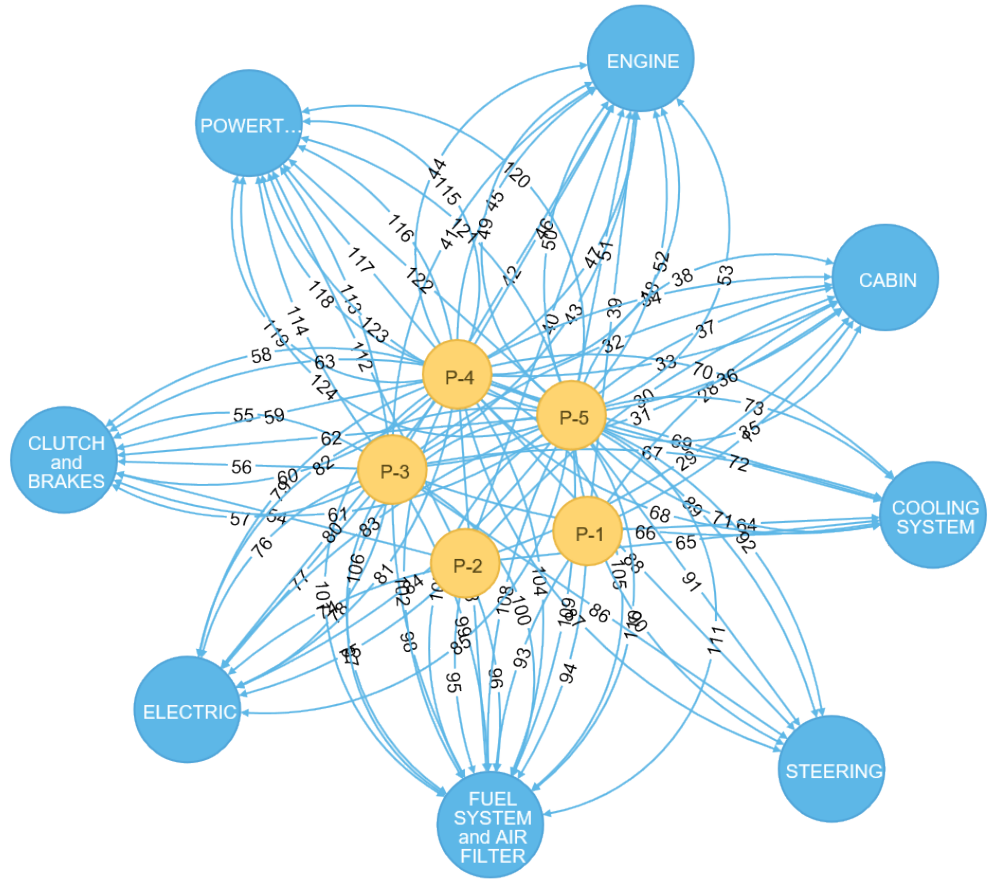
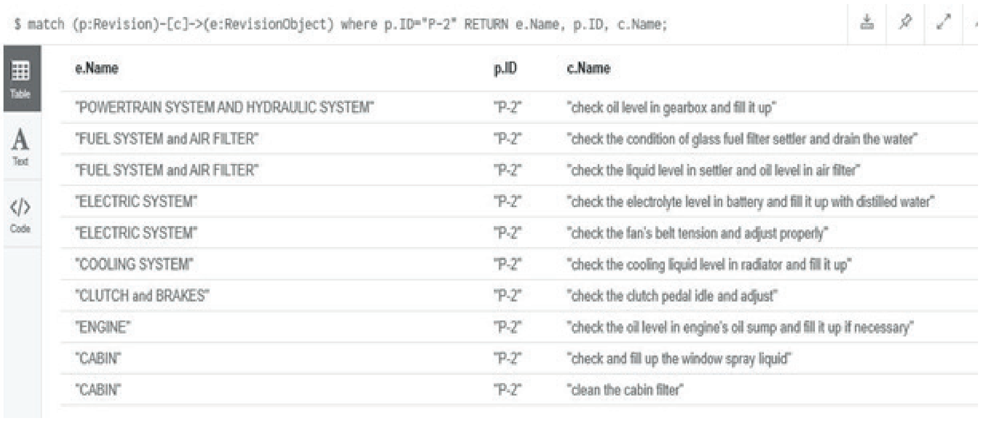


Fig. 7. Result of query Q9: what actions are performed on what revision's object as a path

For some queries the resulting answer is a text, for example: Q10: match (p:Revision)-[c]->(e:RevisionObject) where p.ID="P-2" Return e.Name, p.ID, c,Name; (see the Fig. 8.)



```
$ match (p:Revision)-[c]->(e:RevisionObject) where p.ID="P-2" RETURN e.Name, p.ID, c.Name;
```

e.Name	p.ID	c.Name
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"P-2"	"check oil level in gearbox and fill it up"
"FUEL SYSTEM and AIR FILTER"	"P-2"	"check the condition of glass fuel filter settler and drain the water"
"FUEL SYSTEM and AIR FILTER"	"P-2"	"check the liquid level in settler and oil level in air filter"
"ELECTRIC SYSTEM"	"P-2"	"check the electrolyte level in battery and fill it up with distilled water"
"ELECTRIC SYSTEM"	"P-2"	"check the fan's belt tension and adjust properly"
"COOLING SYSTEM"	"P-2"	"check the cooling liquid level in radiator and fill it up"
"CLUTCH and BRAKES"	"P-2"	"check the clutch pedal idle and adjust"
"ENGINE"	"P-2"	"check the oil level in engine's oil sump and fill it up if necessary"
"CABIN"	"P-2"	"check and fill up the window spray liquid"
"CABIN"	"P-2"	"clean the cabin filter"

Fig. 8. Result of query Q10: what operations and on what revision object should be perform during P-2 revision

Designed GDB can be used for documentation (tracing) revision's history of tractors and then tracking it with appropriate queries. It seems that tracing and tracking process is very important application of GDB

5. Summary

The semantic character of GDB allows to formulate questions regarding internal regularities inherent in specific structures of complex objects. Detection of such regularities using machine learning and inference algorithms provides knowledge enabling effective evolution of existing structures. Having GDB of the entire evolutionary sequence of objects connected to the functional structure, it would be possible to automatically generate future design solutions depending on external scenarios considering the anticipated changes, for example the type of energy sources and operational infrastructure.

Another application of GDB is the possibility of constructive technological development of assemblies and subassemblies of a complex object, considering technical and technological progress based, for example, on replacing the force interaction between elements of the structure with the information effect in order to achieve the desired functionality with a lower energy and cost.

Graph databases can be used in many industries, improving the work of designers, technologists, logisticians, service staff, traders or marketing specialists. Once established and systematically updated, GDB would collect not only the construction and operation data, but also their change in time.

6. References

[1] Chartrand G., Zhang Q. First Course in Graph Theory, Dover Publications (February 15, 2012), ISBN-10: 0486483681, DOI: 10.1007/978-93-86279-39-2.

[2] Chunliu Z., Xiaobing L., Fanghong X., Hongguang B., Kai L. Research on static service BOM transformation for complex products. Advanced Engineering Informatics 36, 2018, 146-162, DOI: https://doi.org /10.1016/j.aei.2018.02.008.

[3] Chłopek Z., Debski B., Szczepański K. Theory and practice of inventory pollutant emission from civilization-related sources. Share of the emission harmful to health from Road Transport, The Archives of Automotive Engineering – Archiwum Motoryzacji, 2018, 79, 1, 5-22, DOI: http://dx.doi. org/10.14669/AM.VOL.79.ART1.

[4] Gajek A. Directions for development of periodic technical inspection for motor vehicles safety systems, The Archives of Automotive Engineering – Archiwum Motoryzacji, 2018, 80, 2, 37-51, DOI: http://dx.doi. org/10.14669/AM.VOL80.ART3.

[5] Gupta S. Neo4j Essentials, Leverage the power of Neo4j is a design, Implement, and Deliver top-notch projects, 2015 Packt Publishing Ltd., ISBN 978-1-78355-517-8.

[6] Harrison G., Next generation databases - NoSQL, NewSQL, and Big Data, Apress 2015, ISBN-13 (electronic): 978-1-4842-1329-2.

[7] Ji Guoli, Gong Daxin, Freddie Tsui, Analysis and implementation of the BOM of a tree-type structure in MRPII, Journal of Materials Processing Technology, Volume 139, Issues 1–3, 2003, Pages 535-538, DOI: https://doi. org/10.1016/S0924-0136(03)00520-X,

[8] Lotko A. Measuring the behavioural loyalty of garage customers, The Archives of Automotive Engineering – Archiwum Motoryzacji, 2018, 79, 1, 37-52, DOI: http://dx.doi. org/10.14669/AM.VOL79.ART3.

[9] Munira Mohd Ali, Rahul Rai, J. Neil Otte, Barry Smith. A product life cycle ontology for additive manufacturing. Computers in Industry, 105, 2019, 191-203, DOI: https://doi.org /10.1016/j.compind.2018.12.007.

[10] Rawia Ahmed Hassan E.L. Rashidy, Peter Hughes, Miguel Figueres-Esteban, Chris Harrison, Coen Van Gulijk. A big data modeling approach with graph databases for SPAD risk, Safety Science, 110, Part B, 2018, 75-79, DOI: https://doi.org/10.1016/j.ssci.2017.11.019.

[11] Robinson I., Webber J.Eifrem E. Graph databases, 2015 Neo Technology, Inc. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 978-1-491-93200-1.

[12] Sasaki B., M., Chao J., Howard R. Graph databases For Beginners, Neo4j EBook. The #1 Platform for connected Date, https://neo4j.com/ (access data June, 11th, 2018).

[13] Sowa J. F. Knowledge Representation: Logical, Philosophical, and Computational Foundations John F. Sowa Pacific Grove, CA: Brooks/Cole, 2000, xiv+594 pp; hardbound, DOI: 10.1162/089120101750300544.

[14] Sowa, J. F. Conceptual graphs as a universal knowledge representation, Computers & Mathematics with Applications, Volume 23, Issues 2–5, 1992, Pages 75-93, DOI: https://doi.org /10.1016/0898-1221(92)90137-7.

[15] Surblys V. Ślaski G. Pikosz H. The usage of a laser height sensors for estimating road unevenness profile, The Archives of Automotive Engineering – Archiwum Motoryzacji, 2018, 79, 1, 5-106, DOI: http://dx.doi. org/10.14669/AM.VOL79.ART7.

[16] Trudeau, R. J. Introduction to Graph Theory, Dover Books on mathematics, Dover Publications; 2nd Edition (February 9, 1994), ISBN-10: 0486678709.

[17] Van Bruggen R. Learning Neo4j, 2014 Packt Publishing LTD.

[18] Van Steen, M.Graph. Theory and Complex Networks; An Introduction, Maarten van Steen (April 5, 2010), ISBN-10: 9081540610.

[19] Agricultural Wheel tractor-URSUS MF-255 with cab -Repair Manual, mechanical plants URSUS, ul. Traktorzystów 1, 02-495 Warszawa, publishing house of Mechanical engineering WEMA, Warszawa, 1989.

[20] Spare Parts Catalogue Tractors Ursus 3512, 3514, plants of tractor industry Ursus, number of publications 70143014M, Warszawa 1995.

[21] Neo4j Graph Academy, https://neo4j.com/graphacademy/ (access data November, 25th, 2018).

[22] Neo4j Download, https://neo4j.com/download/ (access data February, 24th, 2019).

[23] Neo4J Intro to Cypher, https://neo4j.com/developer/cypher-query-language/ (access data November, 25th, 2018).

[24] Neo4J Cypher Refcard 3.4, https://neo4j.com/docs/pdf/neo4j-cypher-refcard-stable.pdf (access data November, 25th, 2018).

[25] The Neo4j Developer Manual v3.4, https://neo4j.com/docs/developer-manual/3.4/ (access data February, 24th, 2019).

[26] The Neo4j Operations Manual v3.4, https://neo4j.com/docs/operations-manual/3.4/ (access data November, 25th, 2018).

7. Supplement

```
//Q1
//Tractor 3514 - models 102, 304, 315
MERGE (c:Tractor {Type:3514, name: 'Ursus 3514'}) MERGE (m:Model {Name:315}) MERGE
(c)-[r:has_a_model]->(m);
MERGE (c:Tractor {Type:3514, name: 'Ursus 3514'}) MERGE (m:Model {Name:304}) MERGE
(c)-[r:has_a_model]->(m);
MERGE (c:Tractor {Type:3514, name: 'Ursus 3514'}) MERGE (m:Model {Name:102}) MERGE
(c)-[r:has_a_model]->(m);
```

```
//Q2
//Eight speed gearbox - modele 102, 304, 315
MERGE (m:Model {Name:315}) MERGE (z:subassembly {Symbol:'7007 027 M91', Name:'Eight speed
gearbox'})  MERGE (m)-[r:has_a_part]->(z);
MERGE (m:Model {Name:304}) MERGE (z:subassembly {Symbol:'7007 015 M91', Name:'Eight speed
gearbox'})  MERGE (m)-[r:has_a_part]->(z);
MERGE (m:Model {Name:102}) MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed
gearbox'})  MERGE (m)-[r:has_a_part]->(z);
```

```
//Q3
Match (c)-[r:has_a_model]->(m)-[z]->(s) RETURN c, r, m, z, s
```

```
//Q4
//Parts of Gearbox Model 102
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'354 441 X1', Name:'Bolt'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'964 167 M1', Name:'Bottom cover'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'354 076 X1', Name:'Split pin'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'886 719 M1', Name:'Clutch fork'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'882 509 M1', Name:'Plug'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part {Symbol:'2
950 X', Name:'Plug'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'881 144 M1', Name:'Screw'})  MERGE (z)-[r:has_a_part {quantity:10}]->(c);
```

```
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180 475 M1', Name:'Wire'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180 449 M1', Name:'Fork shaft'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'894 683 M1', Name:'Plug'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'894 685 M1', Name:'Plug'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1873 532 M1', Name:'Fork, 3rd gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1678 865 Ml', Name:'Rail'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180418 M2', Name:'Dowel'})  MERGE (z)-[r:has_a_part {quantity:4}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'191 221 M1', Name:'Spring'})  MERGE (z)-[r:has_a_part {quantity:4}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1678 867 M1', Name:'Rail, 1st and reverse gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'894 703 M2', Name:'Rail, planet gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180 599 M1', Name:'Wire'})  MERGE (z)-[r:has_a_part {quantity:3}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180 446 M1', Name:'Selector, planet gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'180 765 M1', Name:'Wire'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'897 065 M2', Name:'Fork, planet gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1860 773 M1', Name:'Fork'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1862 868 M2', Name:'Selector, 2nd gear'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1678 866 M1', Name:'Rail'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'354 262 X1', Name:'Bolt'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'353 434X1', Name:'Washer'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1860 761 M1', Name:'Retainer'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1860 764 Ml', Name:'Thrust plate'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1860 767 M1', Name:'Dowel'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'16 777 X', Name:'Ball'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'1860 765 M1', Name:'Packing'})  MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'})  MERGE (c:Part
{Symbol:'353 690 X1', Name:'Bolt'})  MERGE (z)-[r:has_a_part {quantity:2}]->(c);
```

```
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'828 020 MI', Name:'Lock washer'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'184 789 M1', Name:'Stop, reverse gear'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'7006 657 M91', Name:'Gearbox housing'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'184 592 M1', Name:'Bush'}) MERGE (z)-[r:has_a_part {quantity:2}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'180 801 MI', Name:'Bush'}) MERGE (z)-[r:has_a_part {quantity:}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'7006 658 M1', Name:'Gearbox housing'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'180 444 M1', Name:'Shaft, clutch pedal'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'884 691 M1', Name:'Drain plug'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'180 474 M1', Name:'Seal ring'}) MERGE (z)-[r:has_a_part {quantity:1}]->(c);
MERGE (z:subassembly {Symbol:'7007 977 M91', Name:'Eight speed gearbox'}) MERGE (c:Part
{Symbol:'1877 607 MI', Name:'Selector'}) MERGE (z)-[r:has_a_part {quantity:2}]->(c);
```

```
//Q4 What actions are performed to what element as a table
MATCH (p:Revision)-[c]->(e:RevisionObject) RETURN DISTINCT e.Name, c.Name order by e.Name, c.Name;
```

e.Name	**c.Name**
"CABIN"	"change the cabin filter"
"CABIN"	"check and fill up the window spray liquid"
"CABIN"	"check and tighten the cabin fastening screws"
"CABIN"	"clean the cabin filter"
"CLUTCH and BRAKES"	"check the brakes and adjust"
"CLUTCH and BRAKES"	"check the clutch pedal idle and adjust"
"CLUTCH and BRAKES"	"check the work of pneumatic installation and brake valves"
"COOLING SYSTEM"	"check the cooling liquid level in radiator and fill it up"
"COOLING SYSTEM"	"clean the radiator ribs"
"COOLING SYSTEM"	"drain, clean and fill up the cooling system"
"ELECTRIC SYSTEM"	"check the alternator"
"ELECTRIC SYSTEM"	"check the electrolyte level in battery and fill it up with distilled water"
"ELECTRIC SYSTEM"	"check the fan's belt tension and adjust properly"
"ELECTRIC SYSTEM"	"clean the upper battery surface and grease the clamps with technical Vaseline"
"ENGINE"	"change the engine oil"
"ENGINE"	"change the oil in oil sump"
"ENGINE"	"check and adjust the clearance of engine valves"

"ENGINE"	"check the oil level in engine's oil sump and fill it up if necessary"
"ENGINE"	"clean the engine ventilation host"
"FUEL SYSTEM and AIR FILTER"	"change the cartridge of fuel filter"
"FUEL SYSTEM and AIR FILTER"	"check the condition of glass fuel filter settler and drain the water"
"FUEL SYSTEM and AIR FILTER"	"check the liquid level in settler and oil level in air filter"
"FUEL SYSTEM and AIR FILTER"	"clean the cartridge and change the oil in wet air filter"
"FUEL SYSTEM and AIR FILTER"	"drain and clean the fuel tank"
"FUEL SYSTEM and AIR FILTER"	"verify the condition of injectors"
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"change the oil in powertrain system and planetary gears"
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"check and adjust the differential lock pedal"
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"check oil level in gearbox and fill it up"
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"check the oil level in powertrain system"
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"	"clean the transmission oil filter"
"STEERING SYSTEM"	"check the bearing pressure of the front wheel hubs and adjust"
"STEERING SYSTEM"	"check the geometry of the front wheels and adjust"

```
//Q5
MATCH (m:Model {Name:102})-[r:has_a_part]->(g)-[z:has_a_part]->(c) RETURN m,r,g,z,c
```

```
//Q5 what operations are inculed in each revision of each tractor's systems  revision for ID="P-2" as
a table
MATCH (p:Revision)-[c]->(e:RevisionObject) where p.ID="P-2" RETURN e.Name, p.ID, c.Name;

e.Name                                    p.ID  c.Name
"POWERTRAIN SYSTEM AND HYDRAULIC SYSTEM"  "P-2" "check oil level in gearbox and fill it up"
"FUEL SYSTEM and AIR FILTER"              "P-2" "check the condition of glass fuel filter settler and
                                                drain the water"
"FUEL SYSTEM and AIR FILTER"              "P-2" "check the liquid level in settler and oil level in air
                                                filter"
"ELECTRIC SYSTEM"                         "P-2" "check the electrolyte level in battery and fill it up
                                                with distilled water"
"ELECTRIC SYSTEM"                         "P-2" "check the fan's belt tension and adjust properly"
"COOLING SYSTEM"                          "P-2" "check the cooling liquid level in radiator and fill it
                                                up"
"CLUTCH and BRAKES"                       "P-2" "check the clutch pedal idle and adjust"
"ENGINE"                                  "P-2" "check the oil level in engine's oil sump and fill it
                                                up if necessary"
"CABIN"                                   "P-2" "check and fill up the window spray liquid"
"CABIN"                                   "P-2" "clean the cabin filter"
```

```
//Q6: Gearbox structure of all considered Tractor Models
MATCH (n)-[r]-> (m) RETURN n, r, m
```

```
///Q7 What actions are performed on ENGINE revision P-4
MATCH (n:RevisionObject {Name:"ENGINE"})<-[a:Action]-(p:Revision {ID:"P-4"}) RETURN n, a, p;
```

```
//Q8 What actions are performed on ENGINE during each revision?
MATCH (n:RevisionObject {Name:"ENGINE"})<-[a:Action]-(p:Revision) RETURN n, a, p;
```

```
//Q9 what actions are performed on what element as a path
MATCH r=(p:Revision)-[c]->(e:RevisionObject) RETURN r;
```