LATVIJAS UNIVERSITĀTE

EKSAKTO ZINĀTŅU UN TEHNOLOĢIJU FAKULTĀTE

**ZINĀŠANU GRAFU DATU SHĒMU   
IZGŪŠANAS METODES**

BAKALAURA DARBS

Autors: **Arina Fokina**

Studentes apliecības Nr.: af21043

Darba vadītājs: Asoc. prof. Uldis Bojārs

RĪGA 2025

ANOTĀCIJA

Bakalaura darba "Zināšanu grafu datu shēmu izgūšanas metodes" mērķis ir izpētīt zināšanu grafu datu shēmu izgūšanas metodes, koncentrējoties uz SHACL (Shapes Constraint Language) standartu un tā ģenerēšanas rīkiem no RDF (Resource Description Framework) datiem.

Zināšanu grafu datu shēmu iegūšana ietver sistemātisku pieeju, lai identificētu un strukturētu pamatā esošās attiecības un vienības datu kopā. Šis process ir ļoti svarīgs, lai uzlabotu datu sadarbību un atvieglotu dažādas lietojumprogrammas, piemēram, vaicājumu formulēšanu un datu analīzi. Darbā ir izklāstītas galvenās metodoloģijas un ietvari shēmu iegūšanai.

Darbā tiks analizēti esošie SHACL ģenerēšanas rīki,

…

Rezultātā, apkopojot saņemtos datus par rīkiem SHACL shēmu izveidei, plānots

izvērtēt iespējas un metodes SHACL shēmu pārveidošanai formātā, kas ir savietojams ar LU MII izstrādāto ViziQuer rīku.

Darba aktualitāte saistīta ar pieaugošo semantisko datu un zināšanu grafu nozīmi mūsdienu informācijas tehnoloģijās.

**Atslēgas vārdi**:

ABSTRACT

Methods for Extracting Data Schemas from Knowledge Graphs

**Keywords**:

SATURA RADĪTĀJS

APZĪMĒJUMU SARAKSTS

**SHACL** – Shapes Constraint Language datu formas ierobežojumu valoda priekš RDF datiem [https://www.w3.org/TR/shacl/]

**SPARQL** [https://www.w3.org/TR/sparql11-query/]

**RDF** – Resource Description Framework [https://www.w3.org/TR/rdf11-concepts/]

**RDF** grafs – RDF graph) [https://www.w3.org/TR/rdf11-concepts/#dfn-rdf-graph]

**W3C** – World Wide Web Consortium izstrādā tīmekļa standartus, kas ļauj izstrādātājiem veidot programmēšanas risinājumus, ievērojot pieejamības, starptautiskuma, privātuma un drošības principus. [https://www.w3.org/]

**HTML**

**HTTP**

**URI** (vienotais resursu identifikators) ir standarts, kas ierobežots ar latīņu rakstzīmēm.

**UNICODE**

**IRI**

**QSE -** mērogojams formu ieguves rīks, kas palīdz iegūt validējošās formas (SHACL) no lieliem zināšanu grafikiem.

**DBLP**

**DBLP KG**

IEVADS

Nolūks

Bakalaura darba nolūks ir izpētīt zināšanu grafu datu shēmu izgūšanas metodes, apkopot saņemtos datus par rīkiem SHACL shēmu izveidei, izprast atšķirības starp SHACL un ViziQuer, izvērtēt iespējas un metodes SHACL shēmu pārveidošanai formātā, kas ir savietojams ar LU MII izstrādāto ViziQuer rīku.

Darbības sfēra

…

Saistība ar citiem dokumentiem

…

Pārskats

Šis dokuments sastāv no \* nodaļām: …

SHACL iegūšana no RDF datiem:

<https://relweb.cs.aau.dk/qse/>

1. SHACL ģenerējošie rīki

1. teorija

Tīmeklī resursu apraksta ietvars (RDF) ir standarta formāts zināšanu attēlošanai un datu publicēšanai. RDF attēlo informāciju orientētu grafu veidā, kur apzīmētās malas norāda mezglu īpašības. Lai veicinātu efektīvāku piekļuvi un apmaiņu, RDF grafa lietotājam ir svarīgi zināt, kādas īpašības sagaidīt, vai spēt paļauties uz noteiktiem strukturāliem ierobežojumiem, kuriem grafs garantēti atbilst [https://lirias.kuleuven.be/retrieve/778778]. Ar šo nolūku ir veidots **SHACL** (Shapes Constraint Language) - apraksta un validē datu struktūru RDF grafos. SHACL palīdz nodrošināt, ka dati šajā grafā atbilst noteiktiem noteikumiem vai "formām". [https://www.w3.org/TR/shacl/] SHACL ir kā veidne vai shēma, kas norāda: "kā datiem vajadzētu izskatīties. Ja tā nav, tā ir kļūda.".[] To izmanto:

* Datu pārbaudei (validācijai).
* Lietotāja saskarņu ģenerēšanai.
* Datu integrācijai no dažādiem avotiem.
* Automātiska koda ģenerēšanai.

Šajā nodaļā tiek aprakstīti galvenie termiņi analizētas tēmas saprašanai.

* 1. Semantiskais timeklis

Semantiskais tīmeklis (Semantic Web jeb Web 3.0) ir sadarbības kustība, ko vada starptautiskā standartu organizācija W3C. Tas ir esošā tīmekļa paplašinājums, kas paredzēts, lai padarītu informāciju saprotamu ne tikai cilvēkiem, bet arī datoriem, izmantojot W3C noteiktos standartus, piemēram, lai identificētu resursus, un piešķir datiem skaidru nozīmi, kas ļauj datoriem saprast attiecības starp dažādiem informācijas elementiem. [<https://www.w3.org/2001/sw/wiki/Main_Page>]

Atšķirībā no tradicionālā tīmekļa, kur informācija galvenokārt ir **HTML** lapās un paredzēta lasīšanai, semantiskais tīmeklis strukturē datus tā, lai tos varētu automātiski apstrādāt un analizēt.[ <https://www.w3.org/RDF/Metalog/docs/sw-easy>]

Veicinot semantiskā satura iekļaušanu tīmekļa lapās, semantiskā tīmekļa mērķis ir pārveidot pašreizējo tīmekli, kurā dominē nestrukturēti un daļēji strukturēti dokumenti, par "datu tīmekli", mērķis ir padarīt interneta datus mašīnlasāmus.

Semantiskā tīmekļa valodu hierarhiju ilustrē **Semantiskā tīmekļa kaudze** (Semantic Web Stack), kur katrs slānis izmanto un balstās uz zemāk esošajiem. [https://en.wikipedia.org/wiki/Semantic\_Web\_Stack]. Tālāk esošajā attēlā \*.\* ir parādīta semantiskā tīmekļa arhitektūra kā kūka slānis un tā dažādie komponenti:



\*.\* semantiska tīmekļa kaudze [https://www.w3.org/RDF/Metalog/docs/sw-easy]

A diagram of a computer

AI-generated content may be incorrect.

\*.\* semantiska tīmekļa kaudze [https://medium.com/@yassine.hammar1/inight-into-semantic-web-and-why-its-the-next-technological-revolution-24521cec4459]

Virs pamata slāņiem (**URI** un **UNICODE**) ir **RDF** (Resource Description Framework), kas nodrošina datu strukturēšanu, attēlojot informāciju kā saistītus grafus. Augstākajos slāņos ietilpst tādas valodas kā SHACL (Shapes Constraint Language), kas ļauj definēt noteikumus un validēt RDF datus.

OWL standarta secinājumu uzdevums ir dedukcija, savukārt SHACL galvenais uzdevums ir RDF grafu validācija attiecībā pret formu shēmām.

[ <https://www.bartbogaerts.eu/articles/2022/006-LPNMR-SHACL-DL/ShaclDL.pdf>]

[https://spinrdf.org/shacl-and-owl.html]

Šāda pieeja ir būtiska **zināšanu grafiem**, jo tā nodrošina strukturētu datu pamatu, kas ir šī darba pētījuma pamatā.

* 1. Zināšanu grafi

**Zināšanu grafā** (Knowledge Graphs) ir strukturētas datu kopas, kas attēlo informāciju kā savstarpēji saistītus objektus un to attiecības. Zināšanu grafi sastāv no mezgliem, kas apzīmē objektus (piemēram, "Rīga"), un malām, kas norāda attiecības (piemēram, "ir galvaspilsēta" ar "Latvija"). Tos veido, izmantojot semantiskā tīmekļa standartus, piemēram, **RDF** (Resource Description Framework), kas ļauj datus organizēt grafu formā. Vairāk RDF nodaļā. (Sk. \*\*) [https://www.w3.org/TR/rdf11-concepts/#data-model]. Šāda struktūra ļauj integrēt un analizēt datus no dažādiem avotiem, padarot tos noderīgus meklēšanas, datu apstrādes un zināšanu pārvaldības sistēmās.

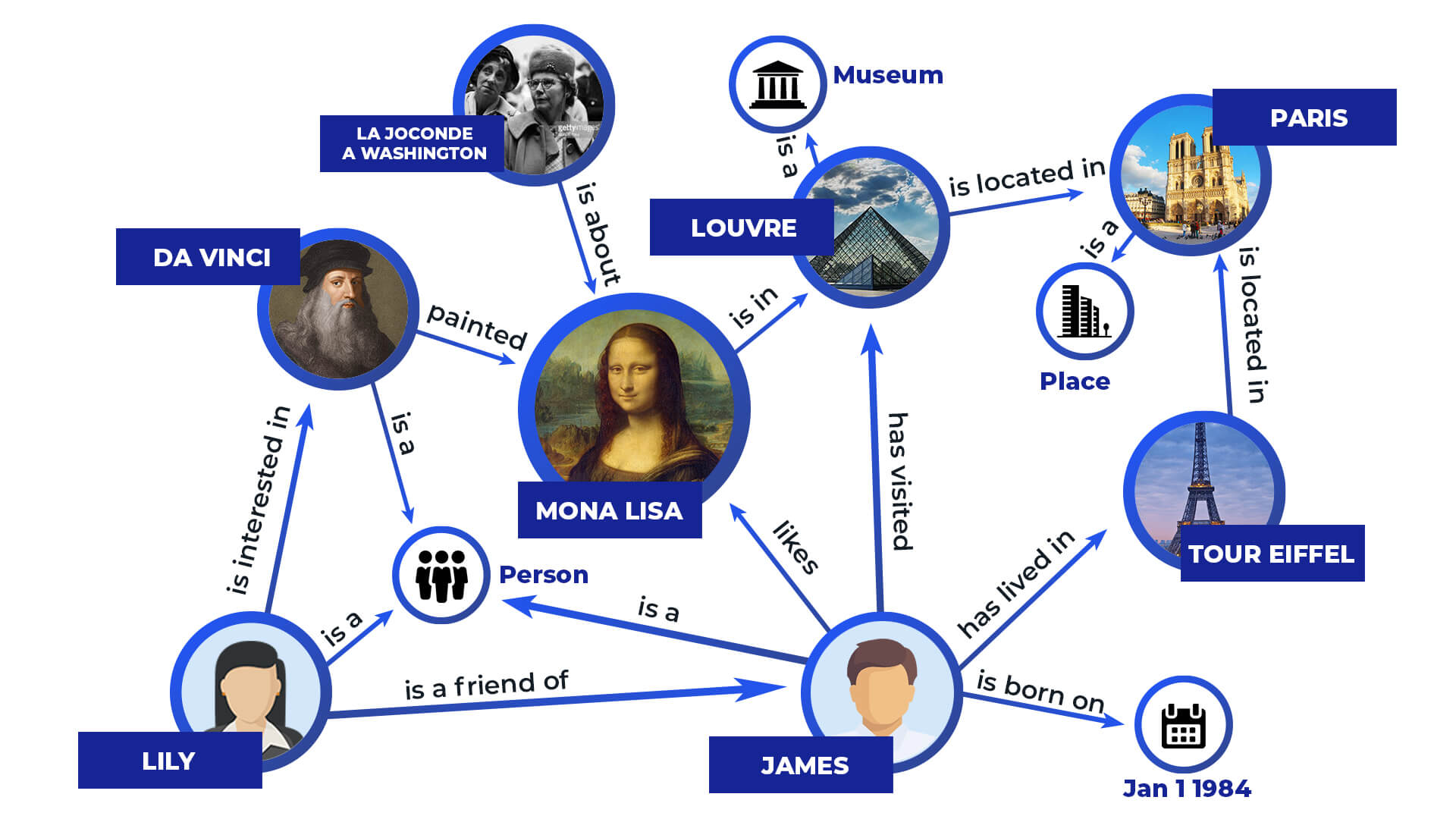
Par termiņu vēl notiek skaidrību definēšana, tās popularitāti nepārprotami ietekmēja Google zināšanu diagrammas ieviešana 2012. gadā, un kopš tā laika šis termins ir plaši izmantots bez definīcijas un tām ir daudz dažādu interpretāciju, vairāk par to šajā tekstā “Towards a Definition of Knowledge Graphs”: https://ceur-ws.org/Vol-1695/paper4.pdf

A close-up of a sign

AI-generated content may be incorrect.

\*.\* Zināšanu grafa arhitektūra [https://ceur-ws.org/Vol-1695/paper4.pdf]

Zināšanu grafi palīdz atrisināt dažas specifiskas problēmas, kas saistītas ar lielajiem datiem.[ <https://www.atulhost.com/what-is-knowledge-graph>]



\*.\* Zināšanu grafa illustrācija[ <https://www.atulhost.com/what-is-knowledge-graph>]

Viena no zināšanu grafiem galvenajām funkcijām ir elastīgums mainīties, jo laika gaitā turpina mainīties plašs datu un metadatu klāsts.

Vēl viena svarīga funkcija ir iespēja lietotājiem veikt datu vaicājumus un atrast atbildes uz jautājumiem datu kopā. Zināšanu grafs var arī sniegt semantisku ieskatu, grafā saglabājot konkrētas datu kopas nozīmi, nevis tikai sniedzot neapstrādātu informāciju bez interpretācijas. [

<https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/>  
<https://www.atulhost.com/what-is-knowledge-graph>

]

A diagram of a diagram

AI-generated content may be incorrect.

\*.\* Zināšanu grafa ilustrācija [<https://medium.com/@yassine.hammar1/inight-into-semantic-web-and-why-its-the-next-technological-revolution-24521cec4459>]

A diagram of shoes and a diagram of a graph

AI-generated content may be incorrect.

\*.\* Zināšanu grafa salīdzinājums ar vienkāršo grafu [https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/]

Zināšanu grafi apvieno vairāku datu pārvaldības paradigmu īpašības:

* **Datu bāze** - datus var izpētīt, izmantojot strukturētus vaicājumus;
* **Grafi** - tos var analizēt kā jebkuru citu tīkla datu struktūru;
* **Zināšanu bāze** - tiem ir formāla semantika, ko var izmantot, lai interpretētu datus un secinātu jaunus faktus.

[https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/]

Ir lielas datu kopas RDF formātā, tos izmanto **semantiskajam tīmeklim,** zemāk ir aprakstīti lielāko zināšanu grafu piemēri**:**

* Google Knowledge Graph - Google padarīja “Knowledge Graph” terminu populāru, 2012. gadā izziņojot savu zināšanu diagrammu. Tomēr ir ļoti maz tehnisku detaļu par tā organizāciju, pārklājumu un lielumu. Ir arī ļoti ierobežoti līdzekļi šīs zināšanu diagrammas izmantošanai ārpus Google projektiem. Sastāv no mezgliem (subjektiem), kas savienoti viens ar otru caur šķautnēm (attiecībām). [Papildus par to https://ceur-ws.org/Vol-1695/paper4.pdf]  
  [https://support.google.com/knowledgepanel/answer/9787176?hl=en]
* Facebook Open Graph Protocol ir Facebook izveidots protokols, kas iedvesmots no Dublin Core, Microformats un RDFa. Saskaņā ar oficiālo vietni tas ļauj jebkurai tīmekļa lapai kļūt par objektu, kas bagātināts Sociālajā grafā – grafs, kas attēlo interneta lietotāju personiskās attiecības. Tas ir modelis vai sociālā tīkla attēlojums, kurā jēdziens "grafs" ir aizgūts no grafu teorijas. Sociālais grafs tiek dēvēts arī par "globālu kartējumu, kas parāda visus cilvēkus un to savstarpējās attiecības".  
  [https://ogp.me/]
* Dbpedia ir Wikipedia publisko datu infrastruktūra lielam, daudzvalodu semantisko zināšanu grafiem. Tas bija projekts, kurā Wikipedia raksta saturs tika pārveidots csv formātā, lai būtu labāka piekļuve zināšanām un varētu tās integrēt zināšanu grafikā.   
  Šis projekts izmanto Wikipedia informācijas kastēm raksturīgo struktūru, lai izveidotu milzīgu datu kopu ar 4,58 lietām [https://wiki.dbpedia.org/about] un ontoloģiju, kas enciklopēdiski aptver tādas entītijas kā cilvēki, vietas, filmas, grāmatas, organizācijas, sugas, slimības utt. Šī datu kopa ir saistīta ar datu kustību. Organizācijām ir bijis nenovērtējami vērtīgi papildināt savas iekšējās zināšanu diagrammas ar miljoniem pūļa avotu entītiju.  
  A diagram of a computer server

  AI-generated content may be incorrect.

\*.\* Pašreizējās DBpedia datu nodrošināšanas arhitektūras ilustrācija [https://www.dbpedia.org/about/]

* Geonames - datu kopas lietotājiem ir piekļuve 25 miljoniem ģeogrāfisku vienību un līdzekļu. [https://www.geonames.org/]
* Wordnet - viena no vispazīstamākajām angļu valodas leksikas datubāzēm, kas sniedz definīcijas un sinonīmus. Bieži izmanto, lai uzlabotu NLP[] un meklēšanas lietojumprogrammu veiktspēju. [https://wordnet.princeton.edu/]
* FactForge - pēc gadiem ilgas pieredzes ziņu publicēšanas nozarē Ontotext izveidoja savu zināšanu grafu par saistītajiem atvērtajiem datiem un ziņu rakstus par cilvēkiem, organizācijām un vietām. Tas ietver datus no iepriekš aprakstītajām zināšanas grafiem, kā arī specializētām ontoloģijām, piemēram, “Financial Industry Business Ontology” (Finanšu nozares biznesa ontoloģiju). [http://factforge.net/]

[[https://medium.com/@yassine.hammar1/inight-into-semantic-web-and-why-its-the-next-technological-revolution-24521cec4459  
https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/](https://medium.com/@yassine.hammar1/inight-into-semantic-web-and-why-its-the-next-technological-revolution-24521cec4459https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/)]

* YAGO-4
* LUBM

1. Resource Description Framework

**RDF** (Resource Description Framework jebResursu Apraksta Ietvars) ir **W3C** standarts informācijas attēlošanai tīmeklī. RDF datu kopas tiek izmantotas, lai organizētu RDF grafiku kolekcijas, un tās ietver noklusējuma grafiku un nulles vai vairāk nosauktu grafiku. [https://www.w3.org/TR/rdf11-concepts/#abstract]

Nodrošinā metadatu apmaiņu - ļauj apvienot datus no dažādiem avotiem, jo tas izmanto standartizētu veidu, kā aprakstīt resursus un to attiecības. RDF atbalsta atvērtus standartus, kas padara to piemērotu dažādu sistēmu un lietotņu integrācijai.

Zināšanu grafi, kas attēlotas RDF, nodrošina vislabāko datu integrācijas, apvienošanas, saistīšanas un atkārtotas izmantošanas sistēmu, jo tās apvieno:

* **Ekspresivitāte**: Semantiskā tīmekļa steka standarti – RDF un **OWL** – ļauj raiti attēlot dažāda veida datus un saturu: datu shēmu, taksonomijas un vārdnīcas, visu veidu metadatus, atsauces un pamatdatus. RDF\* paplašinājums ļauj viegli modelēt izcelsmi un citus strukturētus metadatus.
* **Veiktspēja**: visas specifikācijas ir pārdomātas un pārbaudītas praksē, lai ļautu efektīvi pārvaldīt miljardiem faktu un īpašību grafus.
* **Sadarbspēja**: ir virkne specifikāciju datu serializācijai, piekļuvei (SPARQL protokols galapunktiem), pārvaldībai (SPARQL Graph Store) un federācijai. Globāli unikālu identifikatoru izmantošana atvieglo datu integrāciju un publicēšanu.
* **Standartizācija**: viss iepriekš minētais tiek standartizēts, izmantojot W3C kopienas procesu, lai pārliecinātos, ka tiek izpildītas dažādu dalībnieku prasības — no loģiķiem līdz uzņēmuma datu pārvaldības profesionāļiem un sistēmu operāciju komandām.

RDF grafos informācija tiek attēlota kā **trīskāršu ierakstu kopums** (triples), kur katrs kopums sastāv no:

* **Subjekta** (subject) – resurss, ko apraksta. Ir IRI vai tukšs mezgls
* **Predikāta** (predicate) – īpašība vai attiecība, kas savieno subjektu ar objektu. ir IRI
* **Objekta** (object) – vērtība vai resurss, kas ir saistīts ar subjektu. Ir IRI, burtisks vai tukšs mezgls

Trīskāršu kopuma piemērs: "Artūrs - autors - grāmatas".[ <https://www.w3.org/TR/rdf11-concepts/>]

****

**\*.\*** RDF grafs ar diviem mezgliem (subjekts un objekts) un trīskāršs (triple), kas tos savieno (predikāts) [https://www.w3.org/TR/rdf11-concepts/#data-model

A diagram of a graph

AI-generated content may be incorrect.

**\*.\*** RDF grafa un formu apstiprināšanas piemērs. [https://www.vldb.org/pvldb/vol16/p1023-rabbani.pdf]

Detalizētāk par RDF trīskārša trim sastāvdaļām [https://www.w3.org/TR/rdf11-concepts/#dfn-rdf-triple]:

**IRI, literāļi** un **tukšie mezgli** ir trīs veidu mezgli RDF diagrammā un ir atšķirīgi. Piemēram, http://example.org/ kā virknes literāls nav vienāds ar http://example.org/ kā IRI, ne arī ar tukšu mezglu ar tukša mezgla identifikatoru http://example.org/. [https://www.w3.org/TR/rdf11-concepts/#dfn-rdf-triple]

**IRI** (Internationalized Resource Identifier jeb Internacionalizētais resursu identifikators) - izmanto, lai identificētu resursus (objektus vai to attiecības) RDF grafos (piemēram, "Rīga" vai "galvaspilsēta"). Tas ir Unicode rakstzīmju virkne, kas atbilst RFC 3987 [https://www.ietf.org/rfc/rfc3987.txt]   
Atšķirībā no **URI** (Uniform Resource Identifier jebvienotais resursu identifikators) , kas atbalsta tikai latīņu rakstzīmes, IRI ļauj izmantot plašāku simbolu klāstu.   
Ja IRI ir jāizmanto tur, kur tiek pieņemti tikai URI (piemēram, **HTTP**), tas tiek pārveidots (rakstzīme, piemēram, "ā", tiek kodētas %C4%81=)  
Tiem jābūt absolūtiem (piemēram, http://example.com/riga), nevis relatīviem (kā tikai /riga).  
Piemērs IRI: http://example.com/rīga   
Piemērs URI: http%3A%2F%2Fexample\_com%2Fr%C4%ABga=  
IRI galvenā funkcija ir nodrošināt, lai katrs resurss zināšanu grafā būtu unikāli identificējams. Piemēram, IRI var attēlot subjektu "Rīga" vai attiecību "ir galvaspilsēta".  
[https://www.w3.org/TR/rdf11-concepts/#section-IRIs]

**Literāļi** (Literals) - ir konkrētas vērtības, piemēram, teksta virknes, skaitļi vai datumi, kas attēlo informāciju par resursiem. Katrs literālis sastāv no diviem vai trim elementiem: leksiskās formas, kas ir vērtība tekstā (piemēram, "Rīga" vai "42"), un datu tipa IRI, kas norāda vērtības veidu, piemēram, virkni (*http://www.w3.org/2001/XMLSchema****#string***) vai skaitli (*http://www.w3.org/2001/XMLSchema****#integer***).   
Ja literālis ir teksta virkne ar valodu, tam pievieno **valodas etiķeti**, piemēram, "Rīga"@lv latviešu valodai vai "Riga"@en angļu valodai, un **datu tips** šajā gadījumā ir rdf:langString.  
Literāļi ļauj RDF grafiem saglabāt ne tikai attiecības starp resursiem, bet arī specifiskus datus, kas ir būtiski zināšanu grafu veidošanā. Piemēram, triple "Rīga — iedzīvotāju skaits — 600000" satur literāli "600000" ar skaitlisku datu tipu.   
[https://www.w3.org/TR/rdf11-concepts/#section-Graph-Literal]

**Tukšie mezgli** (Blank Nodes) - ir anonīmi elementi, kas atšķiras no IRI un literāļiem, jo tiem nav unikāla identifikatora. Tie tiek izmantoti, lai vienkāršotu resursu attēlošanu vai to attiecības, kuru konkrētā identitāte nav svarīga.  
Piemēram, trīskāršu kopums "Rīga - ir mājvieta - kādam" var ietvert tukšo mezglu "kādam", kas apzīmē nenosauktu personu vai objektu.

Tukšo mezglu galvenā funkcija ir vienkāršot sarežģītu datu struktūru aprakstu zināšanu grafos, kad nav nepieciešams norādīt precīzu nosaukumu.   
Tie ir lokāli, tas ir, to apzīmējumi (piemēram, \_:b1) ir derīgi tikai konkrētā RDF failā vai sistēmā un netiek saglabāti, pārvietojot datus uz citu vidi. Ja nepieciešama pastāvīga identifikācija, tukšos mezglus var aizstāt ar unikāliem IRI, taču tas parasti nav obligāti. Tukšie mezgli ir svarīgi, jo tie papildina RDF elastību, kas ir būtiska datu shēmu izgūšanai un apstrādei šī darba kontekstā.   
[https://www.w3.org/TR/rdf11-concepts/#section-blank-nodes]

RDF grafos tiek izmantoti **datu tipi** (Datatypes) - izmantoti kopā ar literāļiem, lai precizētu vērtību veidu, piemēram, teksta virknes, skaitļus vai datumus [https://www.w3.org/TR/rdf11-concepts/#section-Datatypes].   
Katrs datu tips ir identificēts ar IRI, piemēram, http://www.w3.org/2001/XMLSchema**#string** teksta virknēm vai http://www.w3.org/2001/XMLSchema**#integer** skaitļiem. RDF galvenokārt izmanto **XML** **Schema** [https://www.w3.org/TR/xmlschema11-2/] definētos datu tipus, kas nodrošina saderību ar citiem standartiem.

Datu tipi nosaka, kā literāļa teksta forma (piemēram, "42") kļūst par konkrētu vērtību (skaitlis 42). Piemēram, datu tipam **xsd:boolean** teksta formas "true" (taisnība) un "1" apzīmē vērtību *true* (taisnība), bet "false" (aplams) un "0" — vērtību *false* (aplams). **Strings** ar valodas etiķeti, piemēram, "Rīga"@lv, izmanto īpašu tipu **rdf:langString**, kas ietver valodas informāciju.

**\*.\* tabula.** xsd:boolean tipa literāļi.

|  |  |
| --- | --- |
| **Literālis** | **Vērtība** |
| <“true”, xsd:boolean> | taisnība |
| <“false”, xsd:boolean> | aplams |
| <“1”, xsd:boolean> | taisnība |
| <“0”, xsd:boolean> | aplams |

* 1. SPARQL

**SPARQL** ir vaicājumu valoda RDF datiem, kas ir līdzīga SQL datu bāzēm. Tas ļauj meklēt datus grafā, norādot nosacījumus (piemēram, "atrast visas pilsētas, kurās iedzīvotāju skaits pārsniedz 500 000").   
[]

https://www.wikidata.org/wiki/Wikidata:SPARQL\_tutorial

https://jena.apache.org/tutorials/sparql.html

1. SHACL

SHACL (Shapes Constraint Language jeb Formu Ierobežojumu Valoda) ir W3C standarta valoda, kas izstrādāta, lai validētu RDF grafus attiecībā pret nosacījumu kopu. SHACL ļauj definēt noteikumus un ierobežojumus, kas nosaka, kā RDF grafiem jāizskatās un kādi dati tiem jāsatur - vai RDF dati atbilst noteiktām shēmām un ierobežojumiem. Piemēram, vai noteiktiem resursiem ir obligāti lauki vai to vērtības atbilst noteiktiem kritērijiem (piemēram, "katrai personai ir jābūt vecumam"). RDF nodrošina datu struktūru, bet SHACL nodrošina mehānismu, lai pārliecinātos, ka šie dati atbilst noteiktiem standartiem un prasībām.

SHACL izteiksmīgums un lietojamība tiek uzlaboti ar dažādām funkcijām un rīkiem, kas atvieglo tā pielietojumu reālās situācijās.

Šī standarta lielākā priekšrocība ir tā elastība un pielāgojamība, kas ļauj precīzi noteikt, kādām struktūrām jāatbilst datiem. SHACL piedāvā dažādus rīkus un funkcijas, kas atvieglo tā lietošanu reālās situācijās, piemēram, datu kvalitātes nodrošināšanai un semantisko tīklu validācijai.  
[https://www.w3.org/TR/shacl/]

A diagram of a triangle with arrows and letters

AI-generated content may be incorrect.

**\*.\*** Grafika piemērs, kas ilustrē SHACL valodas funkcijas. [https://www.bartbogaerts.eu/articles/2022/006-LPNMR-SHACL-DL/ShaclDL.pdf]

Šī neformālā diagramma sniedz pārskatu par dažām galvenajām klasēm SHACL vārdnīcā. Katrs lodziņš apzīmē klasi. Lodziņu saturs zem klases nosaukuma uzskaita dažas no īpašībām, kas var būt šo klašu instancēm, kā arī to vērtību tipus. Bultiņas norāda rdfs:subClassOf tripletus. [https://www.w3.org/TR/shacl/#x2.-shapes-and-constraints]

A diagram of a diagram

AI-generated content may be incorrect.

**\*.\*** Grafika piemērs, kas ilustrē SHACL valodas funkcijas. [https://www.bartbogaerts.eu/articles/2022/006-LPNMR-SHACL-DL/ShaclDL.pdf]

SHACL vārdu krājuma Turtle serializācija satur pilnu SHACL vārdu krājumu (https://www.w3.org/ns/shacl.ttl).

* 1. SHACL Core un SHACL-SPARQL

SHACL specifikācija ir sadalīta SHACL Core un SHACL-SPARQL.

* **SHACL Core** sastāv no bieži nepieciešamajiem līdzekļiem formu, ierobežojumu un mērķu attēlošanai. Visām SHACL implementācijām OBLIGĀTI ir jāievieš vismaz SHACL Core.   
  Pamata funkcionalitāte ar iebūvētiem ierobežojumiem (piemēram, sh:datatype, sh:mincount).
* **SHACL-SPARQL** sastāv no visām SHACL Core funkcijām, kā arī uz SPARQL balstītu ierobežojumu uzlabotajām funkcijām un paplašinājuma mehānisma jaunu ierobežojumu komponentu deklarēšanai.  
  Paplašinājums, kas ļauj rakstīt sarežģītus noteikumus, izmantojot SPARQL vaicājumus.

[https://www.w3.org/TR/shacl/#terminology]

SHACL ietver jēdzienus kā klases (class), tipus (type) un hierarhijas (subclass, superclass), kas palīdz kategorizēt resursus. Piemēram, resurss ar tipu Person var būt klases instance. Šie elementi tiks detalizēti aprakstīti, ja nepieciešams. Vairāk dokumentācija: (<https://www.w3.org/TR/shacl/#terminology>)

SHACL darbojas ar divu veidu grafiem:

* **Formu grafs** (Shapes Graph) - satur noteikumus, kas apraksta, kā datiem jāizskatās.
* **Datu grafs** (Data Graph) - satur faktiskos datus, kas jāpārbauda.

SHACL ir sadalīts divās daļās:

* **SHACL Core** (vai vienkārši **SHACL**): pamatfunkcijas, piemēram, vērtību skaita, datu tipu, skaitļu diapazonu u. c. pārbaude.
* **SHACL-SPARQL**: paplašinātas iespējas, kas ļauj izmantot SPARQL vaicājumus sarežģītākām pārbaudēm.

Kā piemēru, var iedomāties, ja mums ir datubāze ar informāciju par personām. Svarīgi, lai šajā datubāze katrai personai būtu vārds (virkne), vecums (skaitlis no 0 līdz 100), ne vairāk kā viena e-pasta adrese. SHACL ļauj aprakstīt šīs prasības kā "formu" un automātiski pārbaudīt, vai visi ieraksti datubāzē atbilst šiem noteikumiem. Ja kaut kas nav kārtībā (piemēram, kādam ir divas e-pasta adreses vai viņš ir -5 gadus vecs), SHACL ziņos par kļūdām.

Šie noteikumi parasti ir šādā formā: Formai ir jābūt... vai X vērtībām ir jābūt literāļiem vai Visiem tripletu objektiem ar predikātu P ir jābūt IRI. Mezgli, kas pārkāpj kādu no šiem noteikumiem, tiek saukti par **nepareizi veidotiem**. Mezgli, kas nepārkāpj nevienu no šiem noteikumiem, tiek saukti par **labi veidotiem**.

Formu grafs, kas apraksta noteikumus:

@prefix sh: <http://www.w3.org/ns/shacl#> .

@prefix ex: <http://example.org/> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ex:PersonShape

a sh:NodeShape ;

sh:targetClass ex:Person ;

sh:property [

sh:path ex:name ;

sh:datatype xsd:string ;

sh:minCount 1 ;

sh:maxCount 1 ;

] ;

sh:property [

sh:path ex:age ;

sh:datatype xsd:integer ;

sh:minInclusive 0 ;

sh:maxInclusive 150 ;

sh:minCount 1 ;

] ;

sh:property [

sh:path ex:email ;

sh:datatype xsd:string ;

sh:maxCount 1 ;

] .

Šeit tiek aprakstīts sēkojošs:

* **ex:PersonShape** - forma, kas attiecas uz visiem **ex:Person** klases objektiem.
* **sh:targetClass ex:Person** - norāda, ka noteikumi attiecas uz visiem **ex:Person** klases gadījumiem.
* Noteikumi **ex:name** īpašībai:
  + Jābūt virknei (**xsd:string**).
  + Nepieciešama tieši viena vērtība (**minCount 1, maxCount 1**).
* Noteikumi **ex:age** ​​​​īpašumam:
  + Jābūt veselam skaitlim (**xsd:integer**).
  + No 0 līdz 150 (**minInclusive 0, maxInclusive 150**).
  + Nepieciešama viena vērtība (**minCount 1**).
* Noteikumi **ex:email** īpašībai:
  + Jābūt virknei.
  + Ne vairāk kā viena vērtība (**maxCount 1**).

RDF datu grafa piemērs:

@prefix ex: <http://example.org/> .

ex:Alice

a ex:Person ;

ex:name "Alice" ;

ex:age 25 ;

ex:email "alice@example.com" .

ex:Bob

a ex:Person ;

ex:name "Bob" ;

ex:age -5 ;

ex:email "bob@example.com" ;

ex:email "bob2@example.com" .

**ex:Alice** izturēs validāciju: viņa atbilst noteikumiem. **ex:Bob** neizturēs, jo vecums ir -5, kas pārkāpj noteikumu "minInclusive 0" un ir divas e-pasta adreses - pārkāpj noteikumu "maxCount 1".

Turpmākajās sadaļās tiks aplūkoti galvenie SHACL aspekti, tostarp tā validācijas iespējas, izteiksmīgums un praktiskie pielietojumi.

* 1. Īpašības un ceļi

SHACL izmanto **īpašības** (properties) un **ceļus** (paths), lai pārbaudītu RDF grafa datus.   
[https://www.w3.org/TR/shacl/#rdf-terminology]:

**Īpašība** (property) ir IRI, kas darbojas kā predikāts RDF trīskāršu izteiksmēs (piemēram, http://example.com/**hasName**). Resursam (RDF termins, piemēram, tas varētu būt ”Rīga”) ir vērtība konkrētai īpašībai, ja grafā ir trīskāršu izteiksme ar attiecīgo subjektu, predikātu un objektu. Piemēram, trīskāršu izteiksme "Rīga — irNosaukums - 'Rīga'@lv" nozīmē, ka "Rīga"@lv **ir vērtība īpašībai** irNosaukums **resursam** "Rīga".  
SHACL izmanto SPARQL rekvizītu ceļus, lai norādītu, kādus datus pārbaudīt. Piemēram:

Vienkāršs īpašums: hasName - pārbauda tiešo savienojumu (Rīga → "Rīga"@lv).

.

**Īpašības ceļi** (property paths) paplašina īpašību jēdzienu, ļaujot aprakstīt sarežģītākas attiecības starp resursiem, piemēram, "vecāka vārds" (ceļš hasParent/hasName). SHACL izmanto **SPARQL** [https://www.w3.org/TR/sparql11-query/#pp-language] ceļu izteiksmes, lai noteiktu, kādas vērtības ir pieejamas caur šādiem ceļiem grafā.   
SHACL izmanto SPARQL rekvizītu ceļus, lai norādītu, kādus datus pārbaudīt. Piemēram: Ceļš: hasParent/hasName - pārbauda vecāka vārdu (Rīga → Persona → "Anna").\

* 1. Saraksti

**SHACL saraksti** (SHACL Lists) - struktūra, kas attēlo sakārtotus elementu kopumus, palīdz strukturēt datus secīgi. Saraksts var būt IRI vai tukšais mezgls un tiek veidots, izmantojot īpašības **rdf:first** un **rdf:rest**.

Īpašība **rdf:first** norāda uz saraksta pirmo elementu.

**rdf:rest** - uz nākamo saraksta daļu.

Saraksts beidzas ar **rdf:nil**, kas apzīmē tukšu sarakstu un kam nav vērtību nevienai no šīm īpašībām.  
Piemēram, saraksts ar diviem elementiem "sarkans" un "zils" var tikt attēlots šādi:   
1) Pirmais mezgls satur rdf:first = "sarkans" un rdf:rest, kas norāda uz otru mezglu,   
2) Otrais mezgls savukārt satur rdf:first = "zils" un rdf:rest = rdf:nil.   
Sanāca saraksts no diviem elementiem ["sarkans", "zils"].  
Katram mezglam ir tieši viens rdf:first un viens rdf:rest un saraksts nevar būt ciklisks (nevar atsaukties uz sevī ar rdf:rest)

* 1. Formas

Formas (shapes) ir SHACL pamatelements, kas definē noteikumus RDF datiem. Piemēram, forma var prasīt, lai katram resursam ar tipu Person būtu īpašība irNosaukums. Formas nosaka, kādas shēmas tiek izgūtas no grafiem.

* 1. Ierobežojumi

Ierobežojumi (constraints) ir specifiski nosacījumi formās, piemēram, "vērtībai jābūt virknei" vai "skaitlim jābūt lielākam par 0". Tie precizē datu struktūru.

1. Datu SHĒMU IZGŪŠANAs metodes

Datu shēmu izgūšana no zināšanu grafiem ir process, kas strukturē RDF datus, definējot to īpašības un attiecības. Šādas shēmas, piemēram, SHACL (Shapes Constraint Language) formā, ir pamats datu validācijai un tālākai apstrādei.

Pastāv vairāki datu shēmu izgūšanas veidi:

* Manuālā izveide: Eksperti raksta shēmas, balstoties uz datu analīzi, kas nodrošina precizitāti, bet ir laikietilpīga.
* Automātiskā izveide: Ietver statistiskus rīkus, kas analizē datu biežumu (piemēram, QSE [https://relweb.cs.aau.dk/qse/]), vai mašīnmācīšanās metodes, kas meklē datu modeļus.
* Hibrīdās metodes: Apvieno automātisko ģenerēšanu ar manuālu pielāgošanu.

Šajā darbā uzmanība tiek pievērsta shēmu automātiska izveide.

* 1. QSE

Šajā nodaļā tiks pētīta automātiskā SHACL shēmu izgūšana, izmantojot rīku QSE (Quantitative Schema Extraction jeb Kvantitatīvā shēmas iegūšana), kas ģenerē shēmas no RDF datiem. QSE ir mērogojams formu ieguves rīks, kas palīdz iegūt validācijas formas (SHACL) no lieliem zināšanu grafiem. Tā analizē datus un izveido noteikumus (formas), pamatojoties uz statistiku: kuras īpašības bieži tiek atrastas, to veidi, lietošanas biežums utt. Piemērs: ja grafā 90% Personas tipa objektu ir rekvizīts hasName, QSE var piedāvāt SHACL kodu, kur hasName ir obligāts rekvizīts personai.

Rīks piedāvā .jar failu, lai palīdzētu lietotājam viegli iegūt SHACL formas, kur norādītas ievades zināšanu grafs ir .nt formātā (N-Triples formāts, ir uz rindiņām balstīts vienkārša teksta formāts RDF grafika kodēšanai.) [https://www.w3.org/TR/n-triples/] un konfigurācijas failā ir pareizi parametri.  
QSE pieprasa, lai sistēmā būtu instalēta Java, lai palaistu .jar failu.  
 [https://github.com/dkw-aau/qse]

Datu shēmu izgūšanai tiks veikts eksperiments ar QSE rīku, kas automātiski ģenerē SHACL shēmas no RDF datiem. Eksperimenta mērķis ir izanalizēt, kā QSE identificē datu struktūru un kā šīs shēmas var pārveidot ViziQuer formātā. Paredzams, ka QSE palīdzēs efektīvi izgūt shēmas, kas ir pamats tālākai apstrādei ViziQuer vidē.

* + 1. QSE darba sākšana

Tiks izvēlēts vienkāršs RDF datu kopums. Izmantots fails ar .nt paplašinājumu, kurā ir ievadīta informācija par diviem objektiem (Anna un Bob) tipa (rdf:type) Person: <http://example.org/anna> un <http://example.org/bob>, abām ir īpašība vārds (hasName) , vecums (hasAge) ir tikai objektam “Anna”. (Sk. \*.\* attēls).  
A screenshot of a computer

AI-generated content may be incorrect.\*.\* testa faila datu aprakstoša shēma.

Ievades faila saturs:

<http://example.org/anna> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.org/Person> .

<http://example.org/anna> <http://example.org/hasName> "Anna" .

<http://example.org/anna> <http://example.org/hasAge> "25"^^<http://www.w3.org/2001/XMLSchema#integer> .

<http://example.org/bob> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example.org/Person> .

<http://example.org/bob> <http://example.org/hasName> "Bob" .

QSE rīks tiks palaists ar ievades failu .nt formātā un rezultātā iegūts izvades fails .tll formātā ar SHACL shēmu:

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://shaclshapes.org/PersonShape> rdf:type <http://www.w3.org/ns/shacl#NodeShape> ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/hasAgePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/hasNamePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/instanceTypePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#targetClass> <http://example.org/Person> .

<http://shaclshapes.org/hasAgePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 5E-1 ;

<http://shaclshapes.org/support> "1"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:integer ;

<http://www.w3.org/ns/shacl#path> <http://example.org/hasAge> .

<http://shaclshapes.org/hasNamePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#path> <http://example.org/hasName> .

<http://shaclshapes.org/instanceTypePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#path> rdf:type .

Rezultātam ir pievienota statistika, kur redzams kādas īpašības ir vai nav obligāti, to veids un uzticamība (confidence). Shēma apraksta:

* Klasi Person ar mērķi ex:Person (support = 2).
* Īpašību hasName kā obligātu virkni (support = 2, confidence = 1.0, datatype = xsd:string).
* Īpašību hasAge kā neobligātu veselo skaitļi (support = 1, confidence = 0.5, datatype = xsd:integer).

Neveiksmīgas pārvēršanas gadījumā, ģenerē tukšo failu un reģistrē rezultāta statusu .log failā. “MODEL:: DEFAULT - SIZE: 0” ieraksts .log failā nozīme, ka QSE neatrada datus, lai izveidotu likumus, tāpēc modelis ir tukšs. Šīs rezultāts varētu būt, ja ievades RDF dati, piemēram, ir izveidoti nepareizi vai rīks neatrada ievades failu.

* + 1. QSE ar reāliem datiem

Kad QSE darbības procedūra ir saprotama, nepieciešams pārbaudīt darbību ar reāliem RDF datiem. Saņemt reālus RDF datus var ar **DBLP** palidzību – datorzinātņu bibliogrāfija nodrošina atvērtu informāciju par galvenajiem datorzinātņu žurnāliem un darbiem. [https://dblp.org/].

A pie chart with text on it

AI-generated content may be incorrect.

\*.\* sektoru diagrammā ir attēlots dažādu publikāciju veidu sadalījums. [https://dblp.org/statistics/distributionofpublicationtype.html]

A diagram of a document

AI-generated content may be incorrect.

\*.\* DBLP zināšanu grafa shēma. [https://github.com/dblp/kg/wiki/dblp-KG-tutorial]

Izmantot **DBLP KG** - DBLP zināšanu grafu (Sk. attēlu augšā) - lai saņemtu datus no DBLP datubāzes var ar **QLever** palidzību - ir bezmaksas atvērtā pirmkoda programmatūra, kurai ir lietotāja saskarne, kas atvieglo SPARQL vaicājumu rakstīšanu. QLever Engine SPARQL interfeiss ir pieejams pēc adreses https://sparql.dblp.org/, bet iegūtu datus var arī ar API vaicājumiem.

[https://blog.dblp.org/2024/09/09/introducing-our-public-sparql-query-service/].

Mēs strādāsim ar DBpedia datiem, bet, tā kā pilnais DBpedia faila (DBpedia\_SHACL.ttl [https://zenodo.org/records/7598613])) apjoms ir ļoti liels, mēs sāksim ar nelielu apakškopu. Lai ierobežotu datu apjomu, tiek ņemta datu kopa no 100 ierakstiem.

Vaicājuma piemērs, izmantojot **curl**, kurā tiks pieprasīti dati par grāmatam (dblp:Book) ar to nosaukumu (dblp:title), autoriem (dblp:authoredBy) un autoru vārdiem (dblp:primaryCreatorName).

curl -H "Accept: text/turtle" \ -G "https://sparql.dblp.org/sparql" \ --data-urlencode "query=PREFIX dblp: <https://dblp.org/rdf/schema#>   
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>   
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>   
CONSTRUCT {

?book rdf:type dblp:Book .

?book dblp:title ?title .

?book dblp:authoredBy ?author .

?author rdf:type dblp:Person .

?author dblp:primaryCreatorName ?authorName .

}

WHERE {

?book rdf:type dblp:Book ;

dblp:title ?title ;

dblp:authoredBy ?author .

?author rdf:type dblp:Person ;

dblp:primaryCreatorName ?authorName .

}

LIMIT 100" \   
> dblp\_books.ttl

Priekš QSE nepieciešams ievades zināšanu grafs .nt (N-Triples) formātā un konfigurācijas fails config.properties. Dati no DBpedia vaicājuma tika saņemti .ttl (Turtle) formātā ar datiem (pieprasījums piedavā sekojošus formātus application/sparql-results+json, application/sparql-results+xml, application/qlever-results+json, text/tab-separated-values, text/csv, text/turtle, application/octet-stream, . “.ttl” formātu mēs varam pārveidot .nt formātā, kuru saprot QSE rīks.Lai pārveidotu dblp\_books.ttl failu uz dblp\_books.nt failu jāizmanto kādu no ārējiem rīkiem, šajā darbā tiks izmantots python kods ar rdflib bibliotēkas palidzību [https://rdflib.readthedocs.io/en/stable/]**.** dblp\_books.nt fails satur RDF datu kopumu ar informāciju par grāmātiem**,** kur '../dblp\_books.ttl' ir faila .ttl formātā vieta datorā, un '../dblp\_books.nt' ir rezultāta faila nosaukums un vieta**.**

A diagram of a diagram

AI-generated content may be incorrect.

\*.\* Protégé rīkā DBLP .ttl вфег faila ilustrējošā shēma (ierakstu ir 100, tāpēc iluestrācijā nav radīti.).

python kods konvertācijai no .ttl uz .nt:

from rdflib import Graph

g = Graph()

g.parse('../dblp\_books.ttl', format='turtle')

g.serialize('../dblp\_books.nt', format='nt')

Fails atbilst SPARQL pieprasījumam, izskatās pilns un bez liekiem datiem: ir grāmatas, to nosaukumi, autoru attiecības, veidi (grāmata, persona) un vārdi. Nav acīmredzamu kļūdu (piemēram, tukšas vērtības vai nederīgi URI). Visos datos ir informācija un dati nav tukši. Fails ir N-Triples formātā, kā to pieprasa QSE.

Faila satura piemērs ar pamatojošiem komentāriem:

<https://dblp.org/rec/phd/de/Malik2006> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <https://dblp.org/rdf/schema#Book> // grāmata.

<https://dblp.org/rec/phd/de/Malik2006> <https://dblp.org/rdf/schema#title> "Aggregate internet traffic: considerations for the planning of high speed IP networks."   
// grāmatas nosaukums.

<https://dblp.org/rec/phd/de/Malik2006> <https://dblp.org/rdf/schema#authoredBy> <https://dblp.org/pid/00/11175> // grāmatas un autora saite.

<https://dblp.org/pid/00/11175> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <https://dblp.org/rdf/schema#Person> // autors ir persona.

<https://dblp.org/pid/00/11175> <https://dblp.org/rdf/schema#primaryCreatorName> "Sireen Habib Malik" // darba autors.

Lai izceidotu pareizu jaunu failu ar QSE rīku, jāmaina iestatījumus “config.properties” failā QSE projekta mapē. QSE rīks tiks palaists ar ievades failu dblp\_books.nt formātā un rezultātā iegūts izvades fails dblp\_books.tll formātā ar SHACL shēmu:

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://shaclshapes.org/BookShape> rdf:type <http://www.w3.org/ns/shacl#NodeShape> ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/authoredByBookShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/instanceTypeBookShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/titleBookShapeProperty> ;

<http://www.w3.org/ns/shacl#targetClass> <https://dblp.org/rdf/schema#Book> .

<http://shaclshapes.org/PersonShape> rdf:type <http://www.w3.org/ns/shacl#NodeShape> ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/hasAgePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/hasNamePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/instanceTypePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/primaryCreatorNamePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#targetClass> <http://example.org/Person> ;

<http://www.w3.org/ns/shacl#targetClass> <https://dblp.org/rdf/schema#Person> .

<http://shaclshapes.org/authoredByBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#IRI> ;

<http://www.w3.org/ns/shacl#class> <https://dblp.org/rdf/schema#Person> ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#node> <http://shaclshapes.org/PersonShape> ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#authoredBy> .

<http://shaclshapes.org/hasAgePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 5E-1 ;

<http://shaclshapes.org/support> "1"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:integer ;

<http://www.w3.org/ns/shacl#path> <http://example.org/hasAge> .

<http://shaclshapes.org/hasNamePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#path> <http://example.org/hasName> .

<http://shaclshapes.org/instanceTypeBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#in> ( <https://dblp.org/rdf/schema#Book> ) ;

<http://www.w3.org/ns/shacl#path> rdf:type .

<http://shaclshapes.org/instanceTypePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "2"^^xsd:int ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#in> ( <https://dblp.org/rdf/schema#Person> ) ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#in> ( <http://example.org/Person> ) ;

<http://www.w3.org/ns/shacl#path> rdf:type .

<http://shaclshapes.org/primaryCreatorNamePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#primaryCreatorName> .

<http://shaclshapes.org/titleBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://www.w3.org/ns/shacl#or> ( [

<http://shaclshapes.org/confidence> 1.7E-1 ;

<http://shaclshapes.org/support> "17"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#IRI> ;

] [

<http://shaclshapes.org/confidence> 8.3E-1 ;

<http://shaclshapes.org/support> "83"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

] ) ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#title> .

SHACL shēmas apraksta klases dblp:Book (ar obligātām īpašībām title un authoredBy) un dblp:Person (ar īpašību primaryCreatorName).

Saņemtie izvades faili :

* dblp\_books\_QSE\_FULL\_SHACL.ttl: Pilna SHACL shēma.
* dblp\_books.csv: Statistika.
* dblp\_books\_RUNTIME\_LOGS.csv: Izpildes laika žurnāli.
* db\_default: Iespējami papildu datubāzes vai pagaidu faili.

A diagram of a property

AI-generated content may be incorrect.

\*.\* Protégé rīkā DBLP .ttl SHACL faila ilustrējošā shēma.

.log failos redzams ieraksts “MODEL:: DEFAULT - SIZE: 87”, kur “SIZE: 87” norāda, ka QSE datos atrada 87 likumsakarības, kurus potenciāli varētu iekļaut SHACL shēmā. Tas nav klašu skaits (mums ir divas klases: dblp:Book un dblp:Person), bet gan kopējais īpašību, attiecību un citu struktūru skaits datos. Augsta vērtība norāda, ka dati ir sarežģīti, ar daudzām potenciālām īpašībām, bet iegūtā shēma tiek filtrēta pēc atbalsta un ticamības parametriem.

QSE tika konfigurēts ar pruning\_thresholds={(0.1,100),(0.5,500)} parametriem support=100, confidence=0.1 un support=50, confidence=0.5. Parametrs support nosaka minimālo instanču skaitu, kurām jābūt ar īpašību, bet confidence norāda, cik lielai daļai instanču tā jābūt (Vairāk 4.1.3. Atzarošanas sliekšņi (atbalsts un pārliecība) QSE rīkā sadaļā). Šie parametri izrādījās pārāk stingri datu kopai (~100 grāmatas), kā rezultātā īpašība title netika atzīmēta kā obligāta (support=83, confidence=0.83), lai gan tā ir gandrīz visās grāmatās. Turklāt shēmā tika iekļautas nevēlamas īpašības hasName un hasAge no iepriekšējā eksperimenta (test.nt), kas liecina par datu sajaukšanos.

Nākamais solis ir atkārtots eksperiments ar mazāk stingriem parametriem (support=2, confidence=0.1 un support=5, confidence=0.5), lai iekļautu visas obligātās īpašības un tīru “Output” mapi, lai nebūtu datu sajaukšanos.

Lai pārbaudītu, vai dblp\_books\_QSE\_FULL\_SHACL.ttl pareizi validē dblp\_books.ttl, var izmantot brīvpieejāmo tīmekļā vietnes SHACL validatoru (https://shacl-playground.zazuko.com/).

A screenshot of a computer

AI-generated content may be incorrect.

\*.\* zazuko validācijas tīmekļā vietnes rezultāts izveidotiem DBLP datiem un SHACL shēmas .ttl failiem.

* + 1. QSE ar reāliem datiem. Otrais eksperiments

QSE tika konfigurēts ar parametriem support=2, confidence=0.1 un support=5, confidence=0.5, kas ir mazāk stingri nekā iepriekšējie (support=100, confidence=0.1 un support=50, confidence=0.5). Šie parametri tika izvēlēti, lai iekļautu visas svarīgās īpašības nelielā datu kopā (~100 grāmatas, ~98 personas).

Rezultātā tika ģenerēta SHACL shēma, kas apraksta klases dblp:Book un dblp:Person. Klasei Book ir obligāta īpašība authoredBy (support=100, confidence=1.0, minCount 1), kas saista grāmatas ar autoriem, bet īpašība title nav atzīmēta kā obligāta (support=83, confidence=0.83 tekstiem un support=17, confidence=0.17 IRI), lai gan tā ir gandrīz visās grāmatās. Tas var būt saistīts ar datu kļūdu, kur 17 grāmatām title ir kā IRI, nevis teksts. Klasei Person ir obligāta īpašība primaryCreatorName (support=98, confidence=1.0, minCount 1), kas atbilst gaidītajam.

alīdzinājumā ar iepriekšējo eksperimentu (test.nt), kas ietvēra tikai vienu klasi Person ar īpašībām hasName un hasAge, DBLP dati ir sarežģītāki, jo ietver saikni starp Book un Person, demonstrējot QSE spēju apstrādāt strukturālas attiecības. Pēc izvades mapes (Output) tīrīšanas tika novērstas nevēlamas īpašības (hasName, hasAge), kas iepriekš tika iekļautas no test.nt. Lai pilnībā apstiprinātu, ka title ir obligāta, nepieciešams pārbaudīt datus (dblp\_books.nt) un, iespējams, pielāgot SPARQL vaicājumu vai datus.

@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://shaclshapes.org/BookShape> rdf:type <http://www.w3.org/ns/shacl#NodeShape> ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/authoredByBookShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/instanceTypeBookShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/titleBookShapeProperty> ;

<http://www.w3.org/ns/shacl#targetClass> <https://dblp.org/rdf/schema#Book> .

<http://shaclshapes.org/PersonShape> rdf:type <http://www.w3.org/ns/shacl#NodeShape> ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/instanceTypePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#property> <http://shaclshapes.org/primaryCreatorNamePersonShapeProperty> ;

<http://www.w3.org/ns/shacl#targetClass> <https://dblp.org/rdf/schema#Person> .

<http://shaclshapes.org/authoredByBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#IRI> ;

<http://www.w3.org/ns/shacl#class> <https://dblp.org/rdf/schema#Person> ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#node> <http://shaclshapes.org/PersonShape> ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#authoredBy> .

<http://shaclshapes.org/instanceTypeBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "100"^^xsd:int ;

<http://www.w3.org/ns/shacl#in> ( <https://dblp.org/rdf/schema#Book> ) ;

<http://www.w3.org/ns/shacl#path> rdf:type .

<http://shaclshapes.org/instanceTypePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#in> ( <https://dblp.org/rdf/schema#Person> ) ;

<http://www.w3.org/ns/shacl#path> rdf:type .

<http://shaclshapes.org/primaryCreatorNamePersonShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://shaclshapes.org/confidence> 1E0 ;

<http://shaclshapes.org/support> "98"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

<http://www.w3.org/ns/shacl#minCount> 1 ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#primaryCreatorName> .

<http://shaclshapes.org/titleBookShapeProperty> rdf:type <http://www.w3.org/ns/shacl#PropertyShape> ;

<http://www.w3.org/ns/shacl#or> ( [

<http://shaclshapes.org/confidence> 1.7E-1 ;

<http://shaclshapes.org/support> "17"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#IRI> ;

] [

<http://shaclshapes.org/confidence> 8.3E-1 ;

<http://shaclshapes.org/support> "83"^^xsd:int ;

<http://www.w3.org/ns/shacl#NodeKind> <http://www.w3.org/ns/shacl#Literal> ;

<http://www.w3.org/ns/shacl#datatype> xsd:string ;

] ) ;

<http://www.w3.org/ns/shacl#path> <https://dblp.org/rdf/schema#title> .

Mērķa klase: <https://dblp.org/rdf/schema#Book>. Support: 100 - atrasti 100 grāmatu eksemplāri, kas atbilst LIMIT 100 SPARQL vaicājumā. Kļūda, ka 17 grāmatās title ir kā IRI, nevis teksts vēl ir.

* + 1. Atzarošanas sliekšņi (atbalsts un pārliecība) QSE rīkā

Parametri **atbalsts** (support) un **pārliecība** (confidence) tiek izmantoti QSE, lai apgrieztu (pruning) atklātās likumsakarības (patterns) SHACL shēmu ģenerēšanas laikā. Tie nosaka, kuri īpašības un klases tiks iekļautas gala shēmā, filtrējot "vājas" vai nepietiekami apstiprinātas likumsakarības. QSE konfigurācijā tika izmantots pruning\_thresholds parametrispāru saraksts, kur katrs pāris norāda divus parametrus: **confidence** un **support**. Piemēram “pruning\_thresholds={(0.1,100),(0.2,200)}”, kur pirmais pāris norāda, ka 0.1 ir atbalsts un 100 ir pārliecība, un otrais pāris norāda, ka 0.2 ir atbalsts un 200 ir pārliecība . Katrs sarakstā esošais pāris ir atsevišķs slieksnis, ko QSE izmanto, lai filtrētu īpašības, ģenerējot SHACL shēmu.

Parametrs **support** nosaka minimālo instanču skaitu, kurām jābūt ar konkrētu īpašību, lai tā tiktu iekļauta SHACL shēmā. Piemēram, support = 100 nozīmē, ka īpašībai ir jābūt vismaz 100 objektos. Savukārt **confidence** norāda, cik lielai daļai instanču šī īpašība ir jābūt. Piemēram, confidence = 0,1 nozīmē, ka īpašībai ir jābūt vismaz 10 % dotās klases objektu, lai to iekļautu modelī.

QSE pārbauda īpašības (piemēram, nosaukumu, autoru) un iekļauj tās SHACL tikai tad, ja tās atbilst vismaz vienam pārim no pruning\_thresholds. Ja nosaukuma īpašībai ir ticamība=0,15 (15 % grāmatu ir nosaukums) un atbalsts=120 (120 grāmatām ir nosaukums), tā pārsniegs slieksni (0,1 100), jo 0,15 > 0,1 un 120 > 100.

Ja īpašībai ir ticamība=0,05 un atbalsts=50, tā nepārsniegs nevienu no sliekšņiem un tiks izslēgta no shēmas.

Šie parametri tika izvēlēti, lai iekļautu gan obligātās īpašības (piemēram, title, authoredBy), gan retāk sastopamās, ņemot vērā datu kopas mazo apjomu (~100 grāmatas). Alternatīvi tika izmēģināti stingrāki parametri (support=100, confidence=0.1 un support=50, confidence=0.5), taču tie izslēdza daudzas īpašības, jo datu kopā nebija pietiekami daudz instanču, lai izpildītu šos kritērijus.

<https://drops.dagstuhl.de/entities/artifact/10.4230/dblp.rdf.ntriples.2025-04-01>  
Liels .nt datu fails.

* + 1. DBPedia SHACL fails no QSE

No QSE izstrādātiejiem var saņemt automātiski iegūtas SHACL formas, ( DBpedia\_SHACL.ttl [https://zenodo.org/records/7598613]) . Šajā failā ir SHACL formas, kas apraksta DBpedia datu ierobežojumus. Tas sniedz gatavu SHACL formu piemēru, ko var izmantot RDF datu validācijai.

.ttl faila testēšana uz DBpedia datiem.

* + 1. Protégé

A screenshot of a computer

AI-generated content may be incorrect.

1. SHACL konvertēšana uz LU MII ViziQuer formātu

ViziQuer ir rīks vizuālai izpētei un vaicājumu veikšanai RDF datu kopās. Tas ļauj lietotājiem vizualizēt datu shēmu diagrammas un vizuāli konstruēt un izpildīt SPARQL vaicājumus. [https://github.com/LUMII-Syslab/viziquer/wiki]   
  
ViziQuer OBIS izveidotas shēmas piemērs:

<https://github.com/LUMII-Syslab/OBIS-SchemaExtractor/blob/master/build/SampleExtractedSchemaV2.json>  
  
A screenshot of a computer

AI-generated content may be incorrect.

ViziQuer playground:  
<https://viziquer.app/>

A diagram of a country

AI-generated content may be incorrect.

\*.\* ViziQuer playground dažādi vizualizācijas piemēri vaicājumu DBLP datiem [https://viziquer.app/]

Kā ViziQuer izmanto shēmas vizuāliem vaicājumiem

Lai izmantotu DBpedia\_SHACL.ttl programmā ViziQuer SHACL shēmas ir jāpārveido formātā, ko saprot ViziQuer (JSON). Šī darba ietvaros tiek izveidots JavaScript kods SHACL shēmas pārveidošanai uz JSON formātu. Pārbaudi veidot, importējot JSON failu ViziQuer un pārbaudiet validāciju/vizualizāciju.

Šim nolūkam var izmantot OBIS-SchemaExtractor [https://github.com/LUMII-Syslab/OBIS-SchemaExtractor]

OBIS-SchemaExtractor un SHACL atšķirības

SHACL apraksta ierobežojumus (noteiktā shēma), savukārt OBIS apraksta faktisko datu izmantošanu (aprakstošā shēma).

SHACL nesatur statistiku (tripleCount, instanceCount), ko satur OBIS.

SHACL izmanto "formu" jēdzienu, savukārt OBIS tieši apraksta klases un īpašības.

* 1. SHACL konvertēšana uz JSON

SHACL konvertēšana uz JSON (JavaScript Object Notation jeb JavaScript objektu notācija) ietver SHACL definēto strukturālo ierobežojumu tulkošanu formātā, kas ir viegli lasāms un lietojams tīmekļa lietojumprogrammās. Tas ļauj integrēt SHACL validācijas iespējas ar dažādiem datu formātiem, tostarp JSON, kas tiek plaši izmantots tīmekļa datu vaicājumos.

SHACL definē ierobežojumus RDF grafiem, kurus var attēlot kā JSON objektus.

JSON ir noderīgs datu serializācijas un ziņojumapmaiņas formāts. Šī specifikācija definē JSON-LD 1.1 — uz JSON balstītu formātu saistīto datu serializācijai. Sintakse ir izstrādāta, lai to varētu viegli integrēt izvietotajās sistēmās, kas jau izmanto JSON, un nodrošina vienmērīgu jaunināšanas ceļu no JSON uz JSON-LD. Tas galvenokārt ir paredzēts kā veids, kā izmantot saistītos datus tīmekļa programmēšanas vidēs, veidot sadarbspējīgus tīmekļa pakalpojumus un glabāt saistītos datus JSON balstītos krātuves dzinējos. [https://www.w3.org/TR/json-ld/#abstract]

Katru SHACL formu var pārvērst JSON objektā ar tādām īpašībām kā tips, īpašība un mezgla veids, kas atspoguļo SHACL definētos ierobežojumus ["SHACL: A Description Logic in Disguise", 2022].

Piemēram, SHACL formu, kas uzspiež noteiktu īpašību mezglam, var attēlot JSON formātā kā:

{

"type": "Shape",

"property": [

{

"path": "ex:propertyName",

"minCount": 1

}

]

}

ViziQuer saprot JSON struktūru

A diagram of a software company

AI-generated content may be incorrect.

\*.\* ViziQuer datu shēmas izklāsts [https://viziquer.lumii.lv/schema-extractor/VQ\_SerializationFormat.pdf]

Lai konvertētu SHACL shēmu no QSE uz ViziQuer saderīgu formātu, jums ir jāveic šādas darbības: No SHACL shēmas jāizvelk klases un īpašības. Shēmai jāpievieno lietošanas statistika (piemēram, tripleCount, instanceCount), palaižot SPARQL vaicājumus RDF datiem. SHACL struktūra jāpārveido uz OBIS JSON formātu.  
JavaScript skripts, kas paredzēts Turtle formāta ievaddatu apstrādei.

Tas dinamiski iegūst klases, īpašības un ierobežojumus no SHACL specifikācijas.

Papildus tam skripts papildina shēmu ar statistikas datiem, piemēram, instanceCount un tripleCount, izmantojot SPARQL vaicājumus.

Rezultātā tiek veidota OBIS shēma ar laukiem, kas nepieciešami ViziQuer rīkam — piemēram, SuperClasses, ClassPairs, OutgoingProperties un IncomingProperties.

Skripts nodrošina uzticamību, ietverot kļūdu apstrādi un validāciju.

Ar šo domu tika izstrādāts JavaScript kods. Parcel, n3 bibliotēka [https://www.npmjs.com/package/n3]

JavaScript konvertēšanai

…

Rezultāts:

…

* 1. JSON uz ViziQuer

Shēmas importēšana programmā ViziQuer Tools. Izstrādāta programma izveidoja JSON failu ar izvilkto shēmu, šis JSON fails ir jāimportē programmā ViziQuer Tools. Pašlaik, lai importētu shēmu, jāizmanto importētāja moduli no Data Shape Server krātuves.[ <https://github.com/LUMII-Syslab/viziquer-tools>].

Lai importētu JSON failu ar izvilktu shēmu pakalpojumā ViziQuer Tools, izmantojot Data Shape Server (DSS) [https://github.com/LUMII-Syslab/data-shape-server], jāveic darbības, kas aprakstītas GitHub dokumentācijā [https://github.com/LUMII-Syslab/data-shape-server/tree/main/import-generic].

Datu formas serveris: DSS ir ViziQuer aizmugursistēma, kas apstrādā shēmas un nodrošina API to ielādei un lietošanai.

Rezultāti

Bakalaura darba ietvaros …

Secinājumi

IZMANTOTĀ LITERATŪRA UN AVOTI

**Grāmatu avoti:**

1. Agile, Altexsoft, “AGILE PROJECT MANAGEMENT: BEST PRACTICES AND METHODOLOGIES” [Tiešsaiste].

Pieejams: https://www.altexsoft.com/media/2016/04/Agile-Project-Management-Best-Practices-and-Methodologies-Whitepaper.pdf. [Piekļūts 03.12.2023].

1. A. Hunt un D. Thomas, “THE PRAGMATIC PROGRAMMER” [Tiešsaiste].

Pieejams: https://ia802308.us.archive.org/19/items/c-26\_20211009/C26.pdf.

1. “LVS 68:1996 Programmatūras prasību specifikācijas ceļvedis”.

Pieejams: No “DatZ2072-LV : Programminženierija” kursa avotiem.

1. “LVS 72:1996 “Ieteicamā prakse programmatūras projektējuma aprakstīšanai”.

Pieejams: No “DatZ2072-LV : Programminženierija” kursa avotiem.

**Tiešsaistes avoti:**

1. Blazor, “ASP.NET Core Blazor” [Tiešsaiste]. Pieejams: https://learn.microsoft.com/en-us/aspnet/core/blazor/?view=aspnetcore-8.0. [Piekļūts 15.10.2023].
2. GIT, GIT. About” [Tiešsaiste]. Pieejams: https://git-scm.com/about. [Piekļūts 08.09.2023].
3. JavaScript, “MDN WEB Docx. JavaScript” [Tiešsaiste]. Pieejams: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [Piekļūts 08.09.2023].
4. JQuery, “JQuery. About” [Tiešsaiste]. Pieejams: https://jquery.com/. [Piekļūts 08.09.2023].
5. JQuery UI, “JQuery UI. About” [Tiešsaiste]. Pieejams: https://jqueryui.com/about/. [Piekļūts 08.09.2023].
6. MVC, “Overview of ASP.NET Core MVC” [Tiešsaiste]. Pieejams: https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-8.0 [Piekļūts 15.10.2023].
7. QSM, “Software Project Performance Benchmark Tables” [Tiešsaiste]. Pieejams: https://www.qsm.com/resources/qsm-benchmark-tables. [Piekļūts 26.12.2023].
8. Resursu pakas, “Provide localized resources for languages and cultures in an ASP.NET Core app” [Tiešsaiste]. Pieejams: https://learn.microsoft.com/en-us/aspnet/core/fundamentals/localization/provide-resources?view=aspnetcore-8.0. [Piekļūts 15.10.2023].
9. Razor Pages, “Introduction to Razor Pages in ASP.NET Core” [Tiešsaiste]. Pieejams: https://learn.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-8.0&tabs=visual-studio. [Piekļūts 15.10.2023].
10. SQL Server, “What is SQL Server?” [Tiešsaiste]. Pieejams: https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16 [Piekļūts 24.12.2023].
11. SQL Server un ASP, “How to connect a .NET 6 Web App to SQL Server” [Tiešsaiste]. Pieejams: https://medium.com/@mickyangelod/connecting-net-6-web-app-to-sql-server-fccf26ad0aed. [Piekļūts 08.09.2023].
12. W3C, “W3C Web Standards” [Tiešsaiste]. Pieejams: https://www.w3.org/standards/about/. [Piekļūts 08.09.2023].

PIELIKUMI

*1. Pielikums* **Programmatūras koda atkārtoti izmantojamas funkcijas sistēmā.**

Bakalaura darbs “SHACL” izstrādāts LU Eksakto zinātņu un tehnoloģiju fakultātē.

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai un/vai recenzentam uzrādītajai darba versijai.

Autors: **Arina Fokina** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_.01.2025**.

Rekomendēju darbu aizstāvēšanai

Vadītājs: **Asoc. prof. Uldis Bojārs** \_\_\_\_\_\_\_\_\_\_\_\_\_ **\_\_\_.01.2025.**

Recenzents: ***<*** Recenzents ***>***

Darbs iesniegts **\_\_\_.01.2025.**

Bakalaura darbu pārbaudījumu komisijas sekretārs (elektronisks paraksts)