Knowledge and Data Final Project

Group 28: Ashilla, Divabelle, Lisa, Taina

Goal

Sales tracking is important for companies as it helps keep track of growth analysis of their business. It is also helpful for them to look at what kind of games tend to be more successful than others. We can judge this by looking at the sales of a product, as well as how well a video game is rated. Data on video game sales of course can get very big, and sometimes it might be overwhelming for these stakeholders to establish an analysis purely from looking at raw databases. It might take a while to search, understand, and analyze such data. Our project aims to help companies and other stakeholders, such as video game users, to look at interesting data regarding video game sales and ratings through providing a simple visualization. It leads to our question, “How can we visualize data to help stakeholders analyze video games sales efficiently?”

The plan for our project is to collect data about video games as well as their sales and ratings. We think that it is better to take a general analysis about multiple video games from different game developers rather than a specific company. It is because our project can reach more audiences. Also, it will be easier for anyone involved in video games to have a wider knowledge about the topic, which can lead them to have more advantages from it. We will also include relevant information and details of these video games, which relate to the country where the games are sold, the genre, the console type that the games are used, the release year, etc. By collecting any data that relates to those categories and presenting it in a more visually appealing and organized way, we believe that it would be beneficial for the companies and the users to easily interpret such factors.

Stakeholders

One of the potential stakeholders for this analysis are video game companies or developers, as it would be useful for them to analyze their market, particularly for product research and development. It would be beneficial for them to compare which video game genre is particularly popular and has high sales or ratings. Through providing this data analysis, these video game businesses could gain some insight as to what kind of products customers would be interested in as well as what would essentially maximize sales.

Another stakeholder could be video game content creators (i.e. YouTubers, streamers) to identify which type of games are popular among the community and what may be appealing. Other stakeholders may simply include video game enthusiasts or customers who want to observe different games, see what different genres may be appealing or worth trying out, or educate themselves on what video game publishers and developers are there.

The design of the analysis and visualizations

The data from the ontology will be moved to the notebook with a select query. Then the data will be modified for the visualization, for example by counting the average and total sales for all the video games. We will use tables and three different kinds of graphs to visualize the data.
To offer more information about the video games, we will first provide tables containing the most popular and highly rated games alongside their descriptions. This table allows the users to see a trend in game popularity and what these games are about.

Next, a bar chart is used to visualize the information about the ratings and the game console, as well as the genre. The height of the bar represents the rating and the bars are clustered according to the console type/genre. A user can see which game genres or platforms may contribute to a higher average user or critic rating in general.

A scatter plot is used to represent the sales of the video games on the x-axis and the ratings of the games on the y-axis. The colors of the dots are also used to convey information. They tell the genre of the video game. This graph allows the user not only to see the current sales of all games, but also to reason about the effect of the genre of the games in the sales. For example, it can be found out that games from a specific genre often have much higher sales and ratings than other genres. Also, if a company has published a game, they can check whether its sales are at least on the approximate level of other games of the same genre because if that is not the case, they should start looking for the reason why their game sales are not as expected.

The next visualization would be a map of the sales of a video game per country or area where the sales of the video game are expressed by the color of the area. The more intense the color is, the higher the sales are in that area. This map helps the user to find sales patterns. For example, they can find a region where all the video game sales are higher than average and thus, they might want to consider that country as a good place to publish a new game or maybe they would like to continue investigating what makes this country or area different. A user could also find different publishers and developers located on the map, and observe their corresponding number of sales and profit.

Ontologies & Data sources

- **VideoGameOntology by OnToologyUser and dgarijo (Daniel Garijo) on GitHub:** https://github.com/dgarijo/VideoGameOntology/blob/master/GameOntologyv3.owl
  One existing ontology that could be reused to build our ontology is the VideoGameOntology that we were able to find on GitHub. This ontology consists of properties we could make use of, such as releaseDate as well as creator, where its domain and range are also already defined.

- **Schema:** https://schema.org/VideoGame
  Another ontology we could reuse for our project is the schema.org Type 'Video Game.' This ontology also consists of useful properties that we can apply to our main ontology, such as gamePlatform, countryOfOrigin, genre, etc.

Two external datasets will be integrated within the ontology. One of these datasets is the CSV file 'Video game test data' which contains information on different video games. This CSV file is a non-RDF dataset, therefore it should be transformed to RDF-format. The 'Video game test data' dataset contains the total amount of sales per area, Europe for example, the user rating, the title, the genre, the platform and the publisher. The sales per area of a specific type of game in combination with the genre will be used to show which type of games are successful in which areas of the world. The ratings will be grouped by genre, to show what type of games are most appreciated. The 'Video game test data' dataset can be found on: https://www.kaggle.com/andrewolney/videogametestdata

Besides the 'Video game test data' dataset, the SPARQL endpoint DBpedia will be used to select information. DBpedia will be used to obtain the coordinates of the countries/areas listed in the 'Video game test data' dataset. Besides the coordinates of an area, the country of the publisher's headquarters can be obtained along with its coordinates. Combining this information with the 'Video game test data' dataset, a visualization of sales or ratings per area can be made. Also a description of each video game can be derived to give an more in depth overview of the most popular games.

Domain and Scope

The domain of our ontology is video games and their sales. We will be using this ontology to create a data visualization notebook application that presents figures relevant to these sales. We chose this domain as we were able to gain access to multiple datasets on the web which contain the required information. The application is set for the stakeholders to have an efficient presentation tool that combines and organizes information that also expresses the success of video games. On the other hand, the ontology itself can be used as a tool to look through a large amount of knowledge that is available in our domain. A growing interest in video games is the main reason that allows for this type of application to be available in this domain.

Our scope focuses on the ratings and sales of such video games, and which ones are highly rated and popular. It also focuses on various details of our games, such as its genre, description, and we can also look at different publishers and developers. We will also look at the sales of these games per region. We think this type of information is a relevant factor to determining which types of games can be seen as successful. We thought it was rather unnecessary to include details about the players, such as player ID or the characters that are being used in the games. Although it relates to video games, those concepts do not seem to contribute to the factors that affect the increase or decrease in sales so we did not find data about this to add as instances to our classes.

Methodology

While creating our ontology, we used a middle-out approach. A middle-out approach to build an ontology starts with the fundamental concept and expands it with more abstract and specific terms. First we came up with a general idea and then defined our domain and scope by brainstorming about the goal and stakeholders of the ontology. Next we found existing ontologies we could reuse in our own ontology. Then, we defined the classes, object and data properties and the hierarchy. Now we construct our ontology, expand it with the existing ontologies and transform our non-RDF data. We inserted this transformed data into our ontology and combined it with data from a SPARQL endpoint. After running the reasoner we checked for inconsistencies. When we concluded the ontology to be correct, we visualized the data in Python.

Conceptualization

For the conceptualization of our ontology, we decided to build it around the main class *VideoGame*. This modelling decision was made as we wanted to consider all the aspects of individual games, and it is important for our application to account for all its details. Our ontology as a whole contains 23 classes, with 9 object properties that connect these classes to our main class, and 13 data properties that connect it to literals and annotations. A complete visualization of our concept is depicted in **Appendix A**, which shows the complete names of all our classes, subclasses, object and data properties. It also depicts relations between subclasses.

Our main class *VideoGame* has four subclasses: *SinglePlayerGame*, *MultiPlayerGame*, as well as *HighlyRatedGame* and *PopularGame*. *HighlyRatedGame* further includes two subclasses *HighlyUserRatedGame* and *HighlyCriticRatedGame*, depending whether it is an accumulated rating from users or critics. *VideoGame* is also a subclass of the class *Game*.

Our main class asserts all the different object properties (i.e. *hasReleaseYear*, *isPlayedOn*, etc.) onto other different classes (i.e. *Year*, *VideoGameConsole*). *Region* has two subclasses, *Continent* and *Country*. The class *VideoGameCharacter* has two subclasses as well, which are *NonPlayerCharacter* and *PlayerCharacter*. We included the class *VideoGameCharacter* despite the domain of our visualization notebook being focused on the business and sales aspect of such games, as we considered the potential reuse of our ontology where it can be applied to areas that may want to include the art and design aspect of our topic. It would also be useful for projects concerning the link between the artistic features of a game and its sales or ratings.

We also asserted the data properties (*hasSales*, *hasRating, etc.*) again onto our main class, *VideoGame*, in order to insert all the important numbers and literals needed for our ontology. It was a crucial step to include all these different data properties as our approach was to include large amounts of information regarding the individual games. We wanted to integrate descriptions for our games as well, and also included labels for the ease of querying relevant information using SPARQL. Other than that, we also have the classes *VideoGameUser* and *VideoGameCritic* which asserts the data property *hasGivenRating* onto the game ratings which are literals. *VideoGameDeveloper* and *VideoGamePublisher* also asserts the data property hasCoordinates onto literals which contain the coordinates of their headquarters.

Ontology

The ontology consists of the 23 classes, 9 object properties and 13 data properties we have added ourselves in addition to the two existing ontologies we are reusing. The existing ontologies are VideoGameOntology and the part of the Schema-ontology that concerns video games. Specifically, there are 32 classes, 31 object properties and 6 data properties from the VideoGameOntology and 37 classes, 19 object properties and 1 data property from the Schema-ontology. Altogether the ontology has 97 classes, 61 object properties and 17 data properties.

Some of the classes and properties of the ontologies describe the same thing and, thus, they are made owl:equivalentClasses or -Properties. These properties are shown specifically in Table 1. The mappings made between the ontologies are subclass and subproperty relations and other class axioms.

We have also added some class restrictions and property axioms to our ontology. Specifically, we have made the following class restrictions:

- vg:PopularVideoGame owl:equivalentClass [vg:hasSales some xsd:float[>1.0f]]
- vg:VideoGame owl:equivalentClass [vg:hasName some vg:VideoGameName]
- vg:VideoGame owl:equivalentClass [vg:hasGenre some vg:VideoGameGenre]
- vg:VideoGameCritic owl:subClassOf [vg:hasGivenRating some xsd:float]
- vg:HighlyUserRatedVideoGame owl:equivalentClass [vg:hasUserRating some xsd:float[>8.5]]
- vg:HighlyCriticRatedVideoGame owl:equivalentClass [vg:hasCriticRating some xsd:float[>85]]

The first class restriction indicates that all things that have sales of more than 1.0 million are popular video games. This restriction is necessary to determine which video games are the most popular. The minimum value of the sales is determined, so that 10% of the game sales are above the value. This is a necessary and sufficient class restriction.

The second class restriction and the third class restriction tell us that all things that have a video game name and/or a video game genre are instances of the class video game. These restrictions are also necessary and sufficient class restrictions and it allows us to infer that an item is a video game even if we just know that it has a video game genre or/and name.

The fourth class restriction is a sufficient class restriction and it indicates that video game critic is a subclass of things that have given a rating to something. The last two class restrictions define two different classes of highly rated video games. If the rating of the game is higher than 8.5 (or 85), then the game is highly rated. These restrictions are necessary and sufficient.

Other modeling decisions that are important for our ontology are the class and property axioms. The full list of property axioms can be found from Table 2. We have chosen that the properties vg:hasCriticRating and vg:hasUserRating are disjoint properties because they can never be the same thing.

The property vg:isPartOf is transitive because if a video game is a part of a video game series that is part of another video game series, the first video game will also be a part of the second video game series.

Our ontology is mostly focused on the video game business and the other ontologies are used to allow for different future uses of the ontology, which also consider the artistic and play-related parts of video games.

Subject	Property	Object
vg:VideoGame	owl:subClassOf	vg:Game
vg:VideoGame	owl:equivalentClass	[vg:hasGenre some vg:VideoGameGenre]
vg:VideoGame	owl:equivalentClass	[vg:hasName some vg:VideoGameName]
vg:SinglePlayerGame	owl:subClassOf	vg:VideoGame
vg:MultiPlayerGame	owl:subClassOf	vg:VideoGame
vg:PopularVideoGame	owl:equivalentClass	[vg:hasSales some xsd:float[> 1.0f]]
vg:Continent	owl:subClassOf	vg:Region
vg:Country	owl:subClassOf	vg:Region
vg:NonPlayerCharacter	owl:subClassOf	vg:VideoGameCharacter
vg:PlayerCharacter	owl:subClassOf	vg:VideoGameCharacter
vg:VideoGameCritic	owl:subClassOf	[vg:hasGivenRating some xsd:float]
vg:VideoGameUser	owl:subClassOf	foaf:Agent
VideoGameOntology:Character	owl:equivalentClass	vg:VideoGameCharacter
VideoGameOntology:Game	owl:equivalentClass	vg:Game
VideoGameOntology:Genre	owl:equivalentClass	vg:VideoGameGenre
schema:VideoGame	owl:equivalentClass	vg:VideoGame
schema:Game	owl:equivalentClass	vg:Game
schema:VideoGameSeries	owl:equivalentClass	vg:VideoGameSeries
vg:HighlyUserRattedVideoGame	owl:subClassOf	vg:HighlyRatedVideoGame
vg:HighlyUserRattedVideoGame	owl:equivalentClass	vg:hasUserRating some xsd:float[>8.5]
vg:HighlyCriticRatedVideoGame	owl:subClassOf	vg:HighlyRatedVideoGame
vg:HighlyCriticRatedVideoGame	owl:equivalentClass	vg:hasCriticRating some xsd:float[>85]
vg:HighlyRatedVideoGame	owl:subClassOf	vg:VideoGame

Table :. Class restrictions & mappings between ontologies

Subject	Property	Object
VideoGameOntology:hasCharacter	owl:EquivalentProperty	vg:hasCharacter
VideoGameOntology:hasGameGenre	owl:EquivalentProperty	vg:hasGenre
vg:hasUserRating	owl:subPropertyOf	vg:hasRating
vg:hasCriticRating	owl:subPropertyOf	vg:hasRating
vg:hasNASales	owl:subPropertyOf	vg:hasSales
vg:hasEUSales	owl:subPropertyOf	vg:hasSales
vg:hasJPSales	owl:subPropertyOf	vg:hasSales
vg:hasCriticRating	owl:disjointProperty	vg:hasUserRating
vg:hasUserRating	owl:disjointProperty	vg:hasCriticRating
vg:isPartOf	owl:transitiveProperty	

Table 2**:** Property Axioms

Inferences in Protégé

In the Ontology section, we have explained several class restrictions that can be concluded from the external data which resulted in meaningful inferences.

The first meaningful inference we have is on the class PopularVideoGame. The restriction is

vg:PopularVideoGame owl:equivalentClass [vg:hasSales some xsd:float[>1.0f]].

We have a class of vg:VideoGame and every instance under this class has data property of hasEUSales, hasJPSales, hasNASales, and hasOtherSales which points out at an integer value. At first, The reasoner needs to check if any instance has a value greater than 1 million for one of the properties hasEUSales, hasNASales, hasOtherSales or hasJPSales. Since we made these mentioned data properties as subproperties of hasSales, the reasoner concludes that every instance that has those properties must have a data property of hasSales. Therefore, for any video games that have the value of sales that are greater than 1 million, this game must be a type of vg:PopularVideoGame. It can be seen in Figure 2, where the instances for the class PopularVideoGames were empty before the reasoner was turned on. After running the reasoner, any instances that follow the first class restrictions are added.

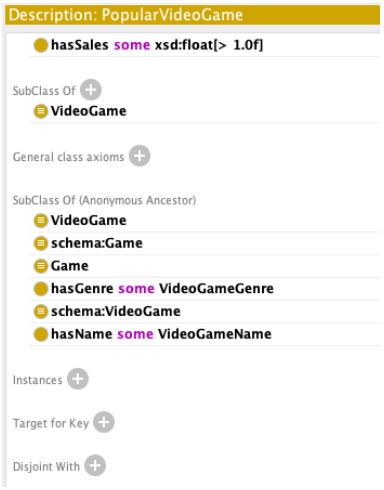


Figure 2.1: vg:PopularVideoGame with reasoner off.

Figure 2.1: vg:PopularVideoGame with reasoner on

The second meaningful inference is on vg:HighlyRatedVideoGames. It is divided into two parts which are vg:HighlyUserVideoGame and vg:HighlyCriticVideoGame.

- vg:HighlyUserRatedVideoGame owl:equivalentClass [vg:hasUserRating some xsd:float[>8.5]]

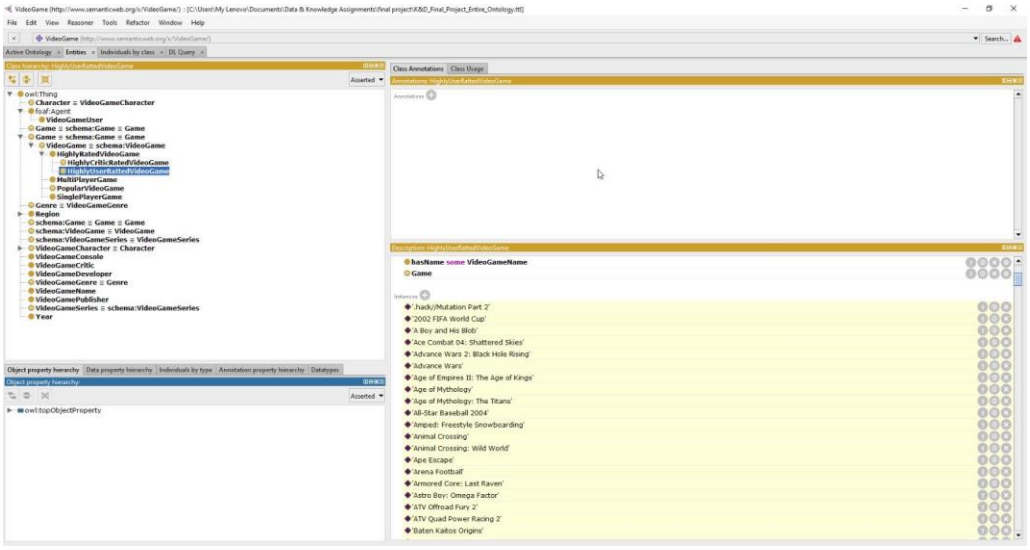


Figure 3: vg:HighlyCriticRatedVideoGame with reasoner on

We know that a particular video game is rdf:type vg:VideoGame and this video game is something that has vg:hasUserRating of some values. Using this class restriction, we can derive that all video games that have the value in the data property vg:hasUserRating that is bigger than 8.5, belong to the class vg:HighlyUserRatedVideoGame.

- vg:HighlyCriticRatedVideoGame owl:equivalentClass [vg:hasCriticRating some xsd:float[>85]]

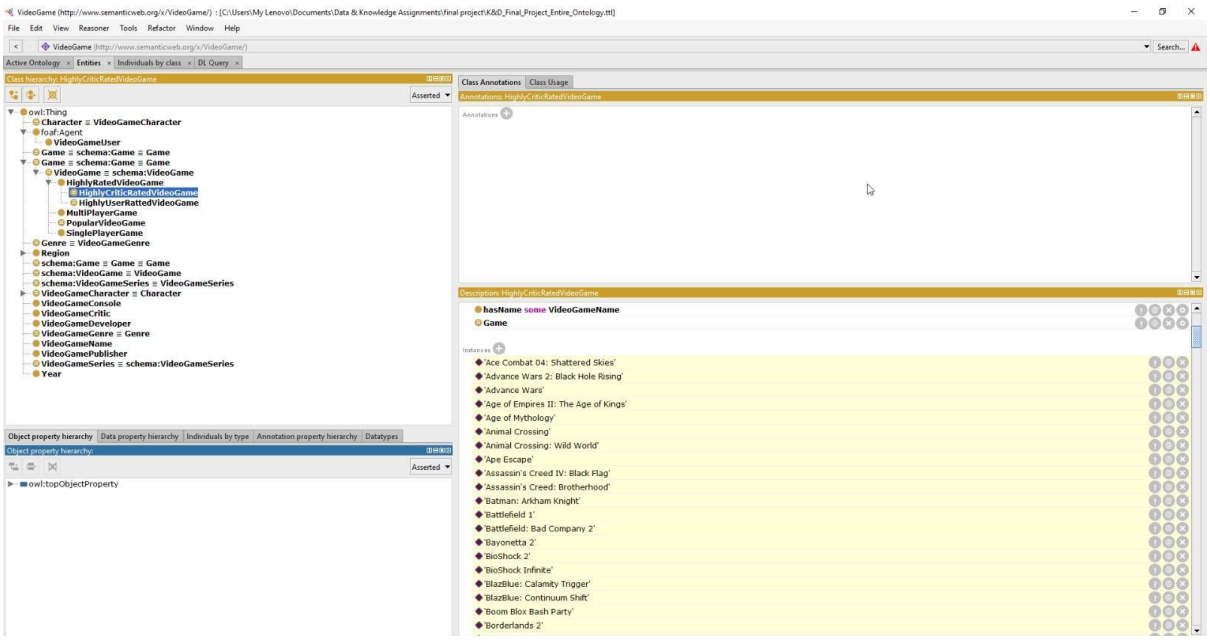


Figure 4: vg:vg:HighlyCriticRatedVideoGame with reasoner on

We know that a particular video game is rdf:type vg:VideoGame and this video game is something that has vg:hasCriticRating of some values. Using this class restriction, we can derive that all video games with the value of their critic rating being bigger than 8.5, belong to the class vg:HighyCriticRatedVideoGame.

We modelled vg:HighlyUserRatedVideoGame and vg:HighlyCriticRatedVideoGame to be subclasses of vg:HighlyRatedVideoGames. Therefore, for every video that are highly user rated and highly critic rated are of the type of highly rated video games.

These class restrictions are meaningful to our application as we will use them to derive a table displaying popular and highly rated games. This would also be useful for the developer and the publisher of a company to simply see how well their game is received by audiences and critics.

Integrating external datasets

We started integrating the data to our ontology by uploading the CSV dataset ‘Videogame test data’ to GraphDB where we mapped the data into the ontology. We mapped the columns Name, Platform, Genre, Publisher, Developer and Year_of_Release of the ‘Videogame test data’ to classes VideoGameName, VideoGameConsole, VideoGameGenre, VideoGamePublisher, VideoGameDeveloper and Year, respectively, in our ontology using rdf:type. Also, the following mappings were made:

- Name vg:hasReleaseYear Year_of_Release
- Name vg:hasNASales Literal(NA_Sales)^^xsd:float
- Name vg:hasEUSales Literal(EU_Sales)^^xsd:float
- Name vg:hasOtherSales Literal(Other_Sales)^^xsd:float
- Name vg:hasJPSales Literal(JP_Sales)^^xsd:float
- Name vg:hasGenre Genre
- Name vg:hasCriticRating Critic_Score
- Name vg:hasUserRating User_Score
- Name vg:isPublishedBy Publisher
- Name vg:isDevelopedBy Developer
- Name vg:isPlayedOn Platform
- Name vg:hasName Name
- Name vg:hasNumberOfCriticRatings Critic_Count
- Name vg:hasNumberOfUserRatings User_Count
- Publisher rdfs:label Literal(PublisherName)@en
- Developer rdfs:label Literal(DeveloperName)@en
- Name rdfs:label Literal(StringName)@en

Also, we declared for all the properties whether they are data or object properties. The rest of the external data is obtained from the SPARQL endpoint of DBpedia. Insert queries were used to obtain data from DBpedia and insert it to our knowledge graph in graphDB. These queries inserted the following triples:

- vg:VideoGame vg:hasDescription ?description .
- vg:Developer vg:hasCoordinates ?geo .
- vg:Publisher vg:hasCoordinates ?geo .

Each of these triples were inserted by executing a different insert query. The query used to insert the ‘vg:VideoGame vg:hasDescription ?description .’ triple is shown below.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX vg: <http://www.semanticweb.org/x/VideoGame/>

INSERT
      {
    GRAPH <http://LAPTOP-RETT404M:7200/repositories/VideoGame> { ?game vg:hasDescription ?desc} } where {
    ?game rdf:type vg:VideoGame;
          rdfs:label ?mylabel .
    optional{
        SERVICE <https://dbpedia.org/sparql> {
            ?s rdfs:label ?mylabel .
            optional{
                ?s rdfs:comment ?desc .
                filter(LANG(?desc) = "en")
}   }   }   }
```

Figure 5: query to insert the description of a video game

This query finds all video games and their labels in the existing graph. Next it sends a request to the DBpedia endpoint. This request searches for a DBpedia instance that has a label that matches one of the video game labels and the english comment of this instance. If such instances and comments exist, the information is inserted as new triples into the knowledge graph. Similar insert queries were used to insert the coordinates of the headquarters of the publishers and developers.

SPARQL queries & inferences

Our knowledge graph now consists of the build ontology, the transformed CSV dataset, the inserted DBpedia data and the inferred triples. In order to visualize the data in Python, the data must be obtained from the knowledge graph saved in graphDB. To do so we used the ipython_sparql_pandas package in Python to send select queries to our graphDB endpoint. Several select queries were sent to the endpoint to receive all data needed to create the visualizations.

One of the visualizations consists of a scatter plot which represents the correlation between the total sales and the user rating of the video games. Also distinction between the genres of the video games is made by using a different color for each genre. To create this scatter plot, for each video game the total amount of sales, the rating and the genre must be received from the graphDB endpoint. This is done by executing the following query:

```
%%sparql http://LAPTOP-RETT404M:7200/repositories/vg2 -qs salesRatingGenre
PREFIX vg: <http://www.semanticweb.org/x/VideoGame/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT distinct ?game ?console ?sales ?userRating ?criticRating ?AVGRating ?genre where {
    ?game rdf:type vg:VideoGame ;
         vg:hasNASales ?NA ;
         vg:hasEUSales ?EU ;
         vg:hasJPSales ?JP ;
         vg:hasOtherSales ?Other ;
         vg:isPlayedOn ?console ;
         vg:hasUserRating ?userRating ;
         vg:hasCriticRating ?criticRating ;
         vg:hasGenre ?genre .
    BIND((?NA+?EU+?JP+?Other) as ?sales) .
    BIND(((?criticRating/10+?userRating)/2) as ?AVGRating) .
}
```

Figure 6: query to select all video games with it sales, rating and genre

The received data is visualized in the following scatter plot:

Figure 7: scatter plot of the correlation between video games sales and rating. Grouped by genre

Another visualization shows an overview of the most popular video games according to their total sales. Here the name of the video game, the total sales and the description of the game are viewed in a table, along with the average rating. The top 10 popular games in descending order are obtained by the following select query:

```
%%sparql http://LAPTOP-RETT404M:7200/repositories/vg2 -qs popularGame
PREFIX vg: <http://www.semanticweb.org/x/VideoGame/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT distinct ?game ?console ?sales ?AVGRating ?description  where {
    ?game rdf:type vg:PopularVideoGame ;
         vg:hasNASales ?NA ;
         vg:hasEUSales ?EU ;
         vg:hasJPSales ?JP ;
         vg:hasOtherSales ?Other ;
         vg:isPlayedOn ?console ;
         vg:hasDescription ?description .
    BIND((?NA+?EU+?JP+?Other) as ?sales) .
    OPTIONAL{?game rdf:type vg:HighlyRatedVideoGame ;
        vg:hasCriticRating ?CriticRating ;
        vg:hasUserRating ?UserRating .
            BIND ((((?CriticRating/10)+?UserRating)/2) as ?AVGRating)}
} order by desc(?sales)
Limit 10
```

Figure 8: query to select the 10 most popular video games

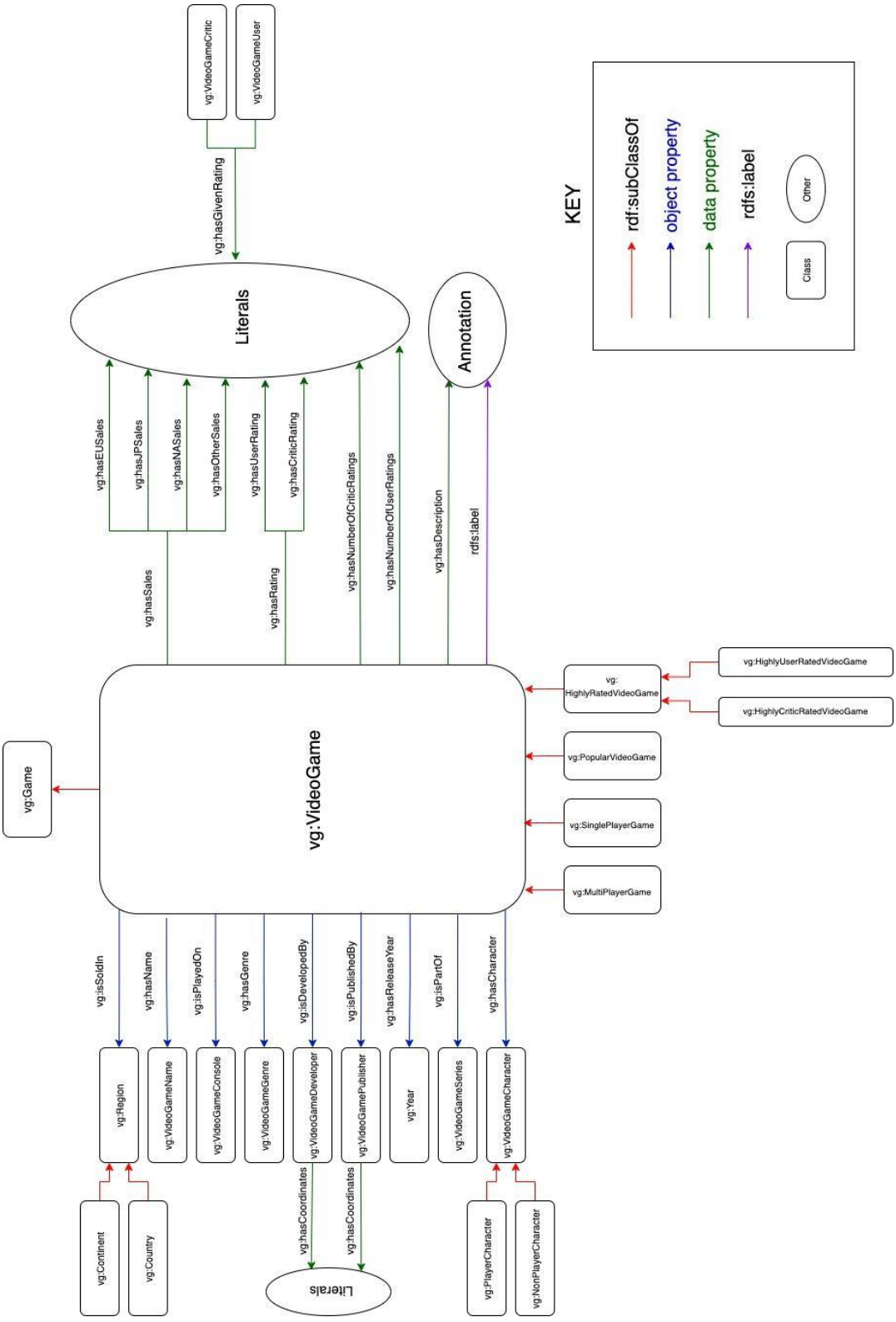
Executing this query results in the following table:

	VideoGame	Console	Sales (x 1 million)	Average rating	Description
0	Mario_Kart_Wii_Wii	Wii	35.520000	NaN	Mario Kart Wii is a 2008 kart racing video game developed and published by Nintendo for the Wii. It is the sixth installment in the Mario Kart series and was released worldwide in April 2008.Like its previous installments, Mario Kart Wii incorporates playable characters from the Mario series, who participate in kart races on 32 different race tracks using specialized items to hinder opponents or gain advantages.
1	Wii_Sports_Resort_Wii	Wii	32.770000	NaN	Wii Sports Resort is a 2009 sports simulation video game developed and published by Nintendo for the Wii video game console, and is a sequel to Wii Sports. It is one of the first titles to require the Wii MotionPlus accessory, which was bundled with the game. Wii Sports Resort was first announced at E3 2008 and was released in Japan on June 25, 2009 and in nearly all other regions in the following month.
2	Wii_Play_Wii	Wii	28.910000	NaN	Wii Play is a party video game developed and published by Nintendo for the Wii console. It was released as a launch game for the console in Japan, Europe, and Australia, and was released in North America in February 2007.
3	Grand_Theft_Auto_San_Andreas_PS2	PS2	20.810000	9.25	Grand Theft Auto: San Andreas is a 2004 action-adventure game developed by Rockstar North and published by Rockstar Games. It is the seventh game in the Grand Theft Auto series, following Grand Theft Auto: Vice City (2002). It was released in October 2004 for PlayStation 2, and in June 2005 for Microsoft Windows and Xbox. The game is set within an open world environment that players can explore and interact with at their leisure.
4	Grand_Theft_Auto_V_X360	X360	16.270000	8.90	Grand Theft Auto V is a 2013 action-adventure game developed by Rockstar North and published by Rockstar Games. It is the first main entry in the Grand Theft Auto series since 2008's Grand Theft Auto IV.
5	Grand_Theft_Auto_Vice_City_PS2	PS2	16.150000	9.10	Grand Theft Auto: Vice City is a 2002 action-adventure game developed by Rockstar North and published by Rockstar Games. It is the fourth main entry in the Grand Theft Auto series, following 2001's Grand Theft Auto III. Set in 1986 within the fictional Vice City, based on Miami, the single-player story follows mobster Tommy Vercetti's rise to power after his release from prison and being caught up in an ambushed drug deal.
6	Call_of_Duty_Black_Ops_II_PS3	PS3	13.789999	NaN	Call of Duty: Black Ops II is a 2012 first-person shooter video game developed by Treyarch and published by Activision. It was released for Microsoft Windows, PlayStation 3, and Xbox 360 on November 12, 2012, and for the Wii U on November 18 in North America and November 30 in PAL regions. Black Ops II is the ninth game in the Call of Duty franchise of video games, a sequel to the 2010 game Call of Duty: Black Ops and the first Call of Duty game for the Wii U.
7	Call_of_Duty_Black_Ops_II_X360	X360	13.679999	NaN	Call of Duty: Black Ops II is a 2012 first-person shooter video game developed by Treyarch and published by Activision. It was released for Microsoft Windows, PlayStation 3, and Xbox 360 on November 12, 2012, and for the Wii U on November 18 in North America and November 30 in PAL regions. Black Ops II is the ninth game in the Call of Duty franchise of video games, a sequel to the 2010 game Call of Duty: Black Ops and the first Call of Duty game for the Wii U.
8	Call_of_Duty_Modern_Warfare_3_PS3	PS3	13.330000	6.00	Call of Duty: Modern Warfare 3 is a 2011 first-person shooter video game, jointly developed by Infinity Ward and Sledgehammer Games and published by Activision. The game was released worldwide in November 2011 for Microsoft Windows, the Xbox 360, PlayStation 3, and Wii. It is the third and final installment in the original Modern Warfare saga, a direct sequel to 2009's Call of Duty: Modern Warfare 2, and the eighth Call of Duty installment overall.
9	Grand_Theft_Auto_III_PS2	PS2	13.100000	9.10	Grand Theft Auto III is a 2001 action-adventure game developed by DMA Design and published by Rockstar Games. It is the first main entry in the Grand Theft Auto series since 1999's Grand Theft Auto 2.

Figure 9: table with the 10 most popular video games, its sales and description

In this table an overview of the most popular games is shown. A popular game is inferred by one of the class restrictions. Namely the restriction: vg:PopularVideoGame owl:equivalentClass [vg:hasSales some xsd:float[>1.0f]] and vg:PopularVideoGame rdf:subClassOf cg:VideoGame. Therefore all video games with a total sales over one million, are also an instance of the class PopularVideoGame. The above select query shows that inferences are made and they can be obtained in Python though the graphDB endpoint.

Appendix A



Appendix A: graph visualization of the VideoGame ontology