Master's Program of Nutrition and Biomedicine

Winter Semester 2022/2023

Bioinformatics internship

Chair of Experimental Bioinformatics (TUM)

Supervisors:

Dr. Josch K. Pauling

Würf,Vivian

Author:

Karim Abdelfattah

Contents:

List of abbreviations:

VMH	Virtual metabolic database
ChEBI	The chemical entities of biological interest
MarkerDB	Marker database
CSV files	Comma separated values files
IntEnz	Integrated Relational Enzyme database
KEGG	Kyoto Encyclopedia of Genes and Genomes
PDBeChem	Dictionary of chemical components
HMDB	Human Metabolome Database

Introduction:

To fully comprehend the pathophysiology of diseases and advance precision medicine, identification of disease-related metabolites is of utmost importance [1]. The main goal is to form as many disease-metabolite connections to build a knowledge graph. The main focus is to clearly associates the different connections between the different chemical compounds, metabolites and disease occurrences. In addition to that is participating in the production of a metabolite network that can be used in literature research to expand on the known metabolite disease connections.

Three databases were used in this endeavor, the virtual metabolic human database (VMH, www.vmh.life) [4], the chemical entities of biological interest (ChEBI) [5], Marker database (MarkerDB) [6].

The VMH gathers data on gut microbial metabolism and connects it to hundreds of diseases and dietary information. Hundreds of carefully curated genome-scale metabolic reconstructions that have been put together using genetic, biochemical, and physiological data form the basis of this database. It also enables complex queries of its content. Additionally, it offers a detailed graphical representation of human metabolism, and disseminates computational models for simulating human and microbial metabolism.

ChEBI provides a large variety of chemical compounds with different identifiers that can be crossed and matched with other databases. To create ChEBI, data from various sources was combined and subjected to merging procedures to eliminate redundancy. The following are four of the primary data sources:

IntEnz is the EBI's Integrated Relational Enzyme database. IntEnz is the Enzyme Nomenclature's master copy [10].

KEGG COMPOUND is a collection of biochemical compound structures that is part of the Kyoto Encyclopedia of Genes and Genomes LIGAND database [11].

PDBeChem - The service that provides web access to the wwPDB's Chemical Component Dictionary as it is loaded into the PDBe database at the EBI [12].

ChEMBL - A database of bioactive compounds, quantitative properties, and bioactivities derived from primary scientific literature [13].

MarkerDB is a freely accessible online database that makes an effort to compile data on all pre-clinical and known clinical biomarkers into a single source. The compiled data enables the finding of the chemical compound disease connections. There are five major types of biomarkers in the

database (condition-specific, protein, chemical, karyotypic, and genetic) and four biomarker categories (diagnostic, predictive, prognostic and exposure). Through the developed search function, users can browse the relevant information by conditions, condition categories, biomarker types, biomarker categories, or search by sequence similarity.

Knowledge graphs have shown to be effective systems for storing and retrieving information [2]. These graphs use nodes and edges to describe the relationships and concepts in biomedicine.[3] Additionally, they yield a visual representation of the relations between different entries such as metabolite disease connections. Numerous biomedical applications can be supported by knowledge graphs. Such Knowledge graphs can help researchers tackle many biomedical problems such as finding new treatments for existing drugs [7], aiding efforts to diagnose patients [8] and identifying associations between diseases and biomolecules [9].

Combining the use of the mentioned databases the goal is to first match as many metabolites as possible between the VMH and ChEBI. Next step is to find the disease connection to the chemical compounds to support the final step of producing a knowledge graph with metabolite- disease connections.

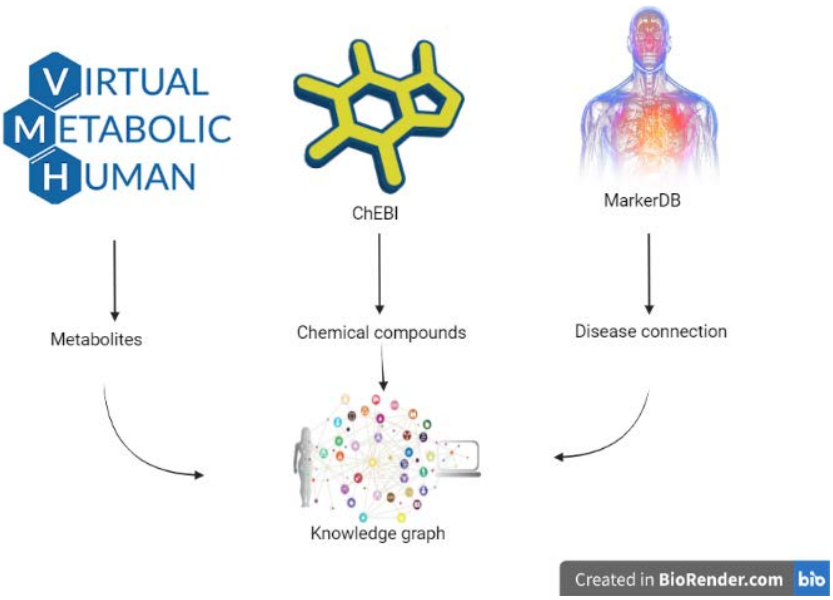

Figure 1: A graphical representation of the work flow.

Methods:

The metabolites and biomarker dataset:

The first task was to obtain all the metabolites available on the VMH database. The VMH database has a very well-built API that one can interact with using the requests package in python and the Pandas library was used to store and clean the data for the matching process [14,15]. The requests package was used while iterating through all the available pages in the VMH database [15]. In addition to that the biomarkers dataset were already matched with disease which was very beneficial to this project as shown in figure 2. Contrary to the metabolite dataset that was missing the disease connections as shown in figure 3. After obtaining all the relevant data, it was stored in comma-separated values (CSV) files for easier access. Last is the cleaning stage such as checking for duplicates, handling null values, renaming columns where needed. That was done using The Pandas library from python [14]. For instance, Column headers were set to lower case, replacing spaces with underscores and renaming the columns to a specific style as shown in figure 2 and 3. The rename function provided by pandas was used in the previous step [14].

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 115 entries, 0 to 114
Data columns (total 24 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   name_of_disease           115 non-null    object
 1   abbreviation_of_disease   115 non-null    object
 2   type_of_disease           115 non-null    object
 3   notes_of_disease          115 non-null    object
 4   phenotype                 115 non-null    object
 5   organ_affected            115 non-null    object
 6   cheblid_of_biomarker      107 non-null    object
 7   description_of_biomarker  115 non-null    object
 8   fullname_of_biomarker     115 non-null    object
 9   abbreviation_of_biomarker 115 non-null    object
 10  inchikey_of_biomarker     115 non-null    object
 11  inchistring_of_biomarker  115 non-null    object
 12  keggid_of_biomarker       81 non-null     object
 13  casregistry_of_biomarker  48 non-null     object
 14  value_of_biomarker        115 non-null    object
 15  normal_conc_of_biomarker  115 non-null    object
 16  ramedis_of_biomarker      115 non-null    object
 17  range_conc_of_biomarker   115 non-null    object
 18  biocyc_biomarker          40 non-null     object
 19  smile_biomarker           115 non-null    object
 20  hmdb_biomarker            115 non-null    object
 21  reconMap3_biomarker       115 non-null    object
 22  food_db_biomarker         66 non-null     object
 23  pubChemId_biomarker       115 non-null    object
dtypes: object(24)
memory usage: 21.7+ KB
```

Figure 2: Biomarker dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5607 entries, 0 to 5606
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   abbreviation_metabolite  5607 non-null   object
 1   casregistry_metabolite   1314 non-null   object
 2   cheblid_metabolite       4765 non-null   object
 3   description_metabolite   5179 non-null   object
 4   fullname_metabolite      5607 non-null   object
 5   hmdb_metabolite          5005 non-null   object
 6   inchikey_metabolite      3044 non-null   object
 7   inchistring_metabolite   5161 non-null   object
 8   keggid_metabolite        4772 non-null   object
 9   biocyc_metabolite        864 non-null    object
 10  food_db_metabolite       1355 non-null   object
 11  reconmap3_metabolite     2349 non-null   object
 12  smile_metabolite         5161 non-null   object
 13  synonyms_metabolite      1828 non-null   object
dtypes: object(14)
memory usage: 613.4+ KB
```

Figure 3: Metabolite dataset

The chemical compounds dataset:

ChEBI was used to generate the chemical compounds dataset. The database was provided as downloadable tab-separated values (TSV) files which were loaded in python again using Pandas then cleaning was performed as the previous cleaning step. Final step was storing the data as csv files as shown in figure 4 [14]. The downloadable files were 4 (compounds,synonym,names and “inchistring”). The matching was done using the internal identifier and the name. The final dataset as shown in figure 4 had all the relevant data combined in one dataset for the matching step.

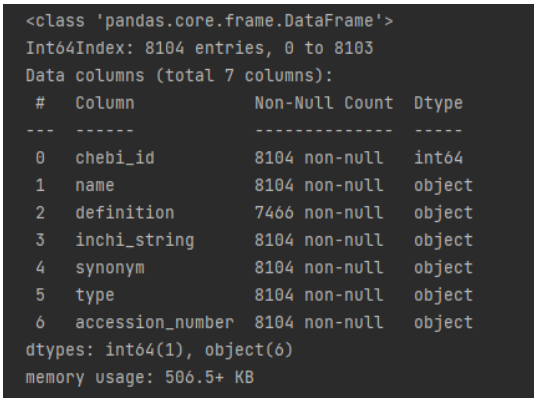

Figure 4: ChEBI dataset.

Matching:

Both previously mentioned dataset had an internal identifier and other identifiers to other databases. The matching was performed based on multiple identifiers to gather as many known metabolites across different databases. The biomarkers dataset was set aside in this stage, only the VMH metabolite and ChEBI dataset were used.

One problem that was encountered was that the same compound was identified multiple times with the internal identifier and that resulted in multiple duplicates in the same column to match the different values in the internal identifier column. To solve this problem, drop_duplicates function- provided by Pandas- was used on specific columns-namely the “name” column- to remove the multiple values while keeping only the first value [14]. This problem was found in the VMH database as the starting number of entries was 5607 and after removing duplicates it is 5502 as shown in figure 5. The 105 removed metabolites were identified in other databases with the same name but for an unknown reason they had a different identifier in the VMH database. That lead

to having the same compound having exact identifiers across different databases but different identifier in the VMH database.

The matching was done using the merge function that is provided by Pandas and an inner match was also used [14].

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5522 entries, 0 to 5606
Data columns (total 14 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   abbreviation_metabolite  5522 non-null   object
 1   casregistry_metabolite   1239 non-null   object
 2   cheblid_metabolite       1509 non-null   float64
 3   description_metabolite   1665 non-null   object
 4   fullname_metabolite      5522 non-null   object
 5   hmdb_metabolite          1689 non-null   object
 6   inchikey_metabolite      2991 non-null   object
 7   inchistring_metabolite   2937 non-null   object
 8   keggid_metabolite        1837 non-null   object
 9   biocyc_metabolite        839 non-null    object
 10  food_db_metabolite       1319 non-null   object
 11  reconmap3_metabolite     2302 non-null   object
 12  smile_metabolite         3221 non-null   object
 13  synonyms_metabolite      1451 non-null   object
dtypes: float64(1), object(13)
memory usage: 647.1+ KB
```

Figure 5: Metabolite dataset after duplicate removal

Matching for the disease information:

The final dataset was used with the MarkerDB file that was provided by the chair. Unnecessary information was removed from this dataset, for instance this data base had columns about the bonds in certain compounds and other information which was not relevant to this matching. The matching was done using the name column. In addition to that, from the downloadable file from the MarkerDB, the disease and publication information could be further matched. The final dataset contained the name of each compound, disease related to it, which publication it was used, at what concentration the disease occurs, the location of the disease effect in the body and finally the gender it affects.

Knowledge graph generation:

For the knowledge graph, a scheme of the nodes along with the relationships had to be made first as shown in figure 6. The used tool is AuraDB which is a graph visualization tool developed by the Neo4j team. It enables simply the uploading of a CSV file and defining from the data in the CSV file, the nodes and edges. There is also the possibility to preview the graph and to run the import. Running the import enables the possibility of querying the graph using Cypher programming language. The nodes were taken from the columns of the final MarkerDB dataset and the edges describe the relationship between the nodes. [16,17]

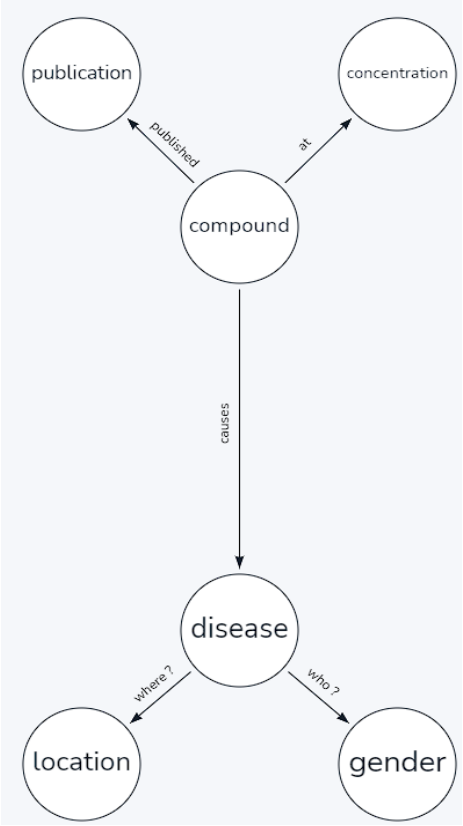

Figure 6: Knowledge graph schematic

Results:

The first matching step:

The starting entries in the VMH were 5502 while the ChEBI dataset were 8104.The first matching was on the “inchistring” column and that yielded 163 entries. The ChEBI dataset was then filtered on 3 different identifiers (CAS_registry, KEGG, HMDB, synonym). Matching was then performed between each of the new dataset to the same column on the VMH dataset. It yielded 23, 116,11

and 1 respectively. Other merging was done also on the name and abbreviation but it did not yield any results. Finally, all of the newly matched datasets were concatenated together using the concatenate function and then cleaned again to avoid any duplications across different identifiers.

The final dataset of metabolites consisted of 257 as shown in figure 7 [14].

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 257 entries, 0 to 7
Data columns (total 21 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   abbreviation_metabolite  257 non-null    object
 1   casregistry_metabolite   151 non-null    object
 2   cheblid_metabolite       179 non-null    float64
 3   description_metabolite   191 non-null    object
 4   fullname_metabolite      257 non-null    object
 5   hmdb_metabolite          193 non-null    object
 6   inchikey_metabolite      234 non-null    object
 7   inchistring_metabolite   238 non-null    object
 8   keggid_metabolite        194 non-null    object
 9   biocyc_metabolite        78 non-null     object
 10  food_db_metabolite       156 non-null    object
 11  reconmap3_metabolite     156 non-null    object
 12  smile_metabolite         230 non-null    object
 13  synonyms_metabolite      182 non-null    object
 14  chebi_id                 257 non-null    int64
 15  name                     257 non-null    object
 16  definition               231 non-null    object
 17  inchi_string             257 non-null    object
 18  synonym                  257 non-null    object
 19  type                     257 non-null    object
 20  accession_number         257 non-null    object
dtypes: float64(1), int64(1), object(19)
memory usage: 44.2+ KB
```

Figure 7: Final dataset (metabolites and ChEBI merge)

The second matching step:

Matching with the MarkerDB yielded only 38 entries in the end which was less than the expected as shown in figure 8. To get the publication and disease information the downloadable file was used from the MarkerDB website. The file provided by the chair had the internal MarkerDB identifier while the downloadable file had the publication and disease information. Matching the final dataset of the metabolites first with the provided file added the internal identifier and using the internal identifier that is also in the downloadable file, the publication and disease information could be added as columns. After matching the final dataset had 38 entries with some null values. The concentration column was duplicated as the AuraDB tends to shift the information under each

column. The first column would have zero values and last column would have the data of the previous to last column.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38 entries, 0 to 37
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   name            38 non-null     object
 1   chebi_id        38 non-null     int64 
 2   mdbid           38 non-null     object
 3   disease         38 non-null     object
 4   markerdb_id     38 non-null     int64 
 5   Gender          38 non-null     object
 6   location        38 non-null     object
 7   concentration   38 non-null     object
 8   concentration_1  38 non-null     object
dtypes: int64(2), object(7)
memory usage: 2.8+ KB
```

Figure 8: Final MarkerDB dataset

Knowledge graph:

After defining the nodes and edges as mentioned before. The import was run, a detailed version of the graph could be observed as shown in figure 9. The graph had 124 nodes and 121 edges. 33 causes, 33 compounds,33 locations and 27 publications. Although 38 compounds were identified in the final dataset only 33 compounds had edges with disease. The reason for that is unknown. Furthermore, a closer look at the graph clearly identifies a compound with all the relevant information. As the 5-Hydroxyindoleacetic acid is linked to carcinoid tumors at 5.0 umol/mmol creatinine with the relevant publication that proves this information as shown in figure 10.

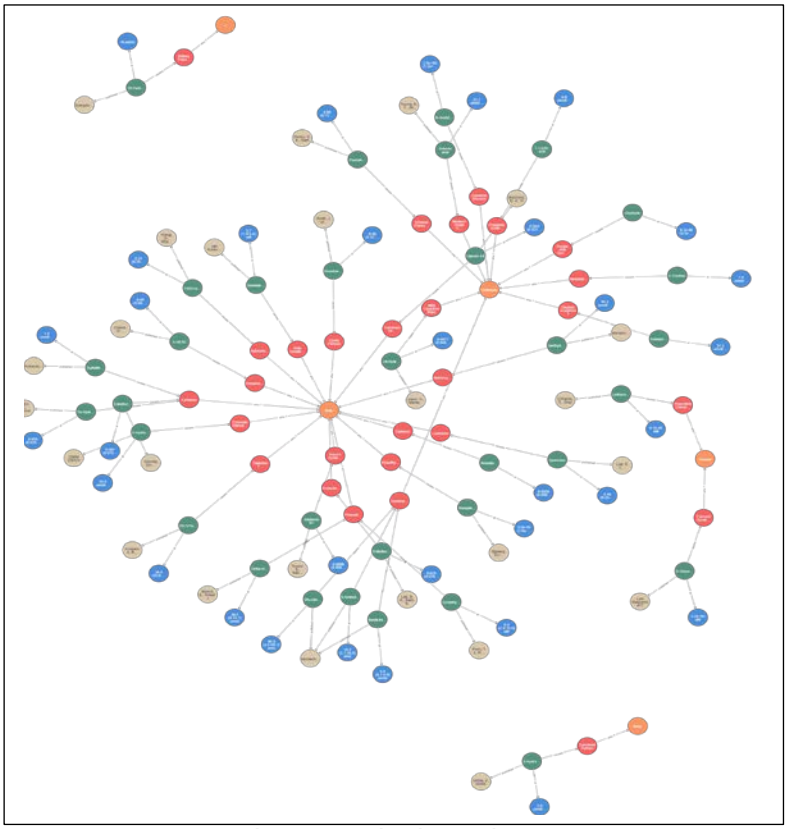

Figure 9: Final graph

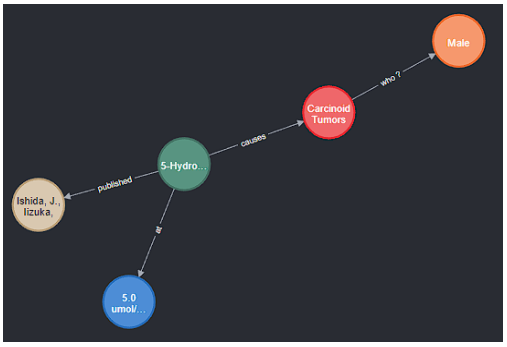

Figure 10: 5-Hydroxyindoleacetic acid Graph

Discussion:

Matching between different databases proved to be daunting task. Contrary to expectations the final matched metabolites were a lot less than the starting entries from the different datasets as explained before. The reason for that could be the different focus of the different databases.

As mentioned before the ChEBI database is formed of different sources focusing on the small chemical compounds [4]. While the VMH database focuses mainly on the disease, gut health, nutrition and human metabolism [5]. In this case it is understandable that a fraction of the metabolites could be matched, as ChEBI focuses on enzymes and small chemical compounds in general in the human metabolism while VMH focuses on the metabolites that are concerned with the microbiota and disease in the context of human metabolism. On the other hand, the Human metabolome database (HMDB) could yield better matching results as it focuses on small chemical compounds but in the context of metabolomics, clinical chemistry and biomarker discovery [18].

MarkerDB on the other hand focuses on identified biomarkers that can predict or diagnose disease. In this regard it stands to reason that not all the metabolites are identified as biomarker in this dataset [6]. On the other hand, data from PubChem would be superior as it is an open chemistry database that has literature citations and toxicity information which could yield better metabolite-disease connections [19].

Conclusion:

Neo4j graph visualization yielded promising results on how to understand the data. For future prospects a better approach is to load the CSV file in Neo4j, create nodes and edges manually and storing the output in a graph object. The process to creating nodes and edges was started, but due to lack of time was never finished. In addition to that the compounds and disease connection in this step were 38 which solved having only 33 in AuraDB. Furthermore, applying

centralization algorithms to find the most frequently occurring nodes would allow further and deeper interpretation of the knowledge graph. For instance, finding the frequently occurring node or the shortest path. In conclusion, matching metabolites across different databases proved to be a challenging task. Acquiring the data when interacting with an API depends on how well the API is designed which is not the case in every database. Merging on identifiers can cause duplications when same compounds are identified multiple times differently. Introducing multiple databases that have the same focus could be a viable solution. The merging process should be done bearing in mind that errors may emerge that were not expected to happen.

References:

1. Wang, Y., Juan, L., Peng, J., Wang, T., Zang, T., & Wang, Y. (2022). Explore potential disease related metabolites based on latent factor model. BMC genomics, 23(Suppl 1), 269.

2. MacLean F. (2021). Knowledge graphs and their applications in drug discovery. Expert opinion on drug discovery, 16(9), 1057–1069.

3. Nicholson, D. N., & Greene, C. S. (2020). Constructing knowledge graphs and their biomedical applications. Computational and structural biotechnology journal, 18, 1414–1428.

4.Noronha, A., Modamio, J., Jarosz, Y., Guerard, E., Sompairac, N., Preciat, G., ... & Thiele, I. (2019). The Virtual Metabolic Human database: integrating human and gut microbiome metabolism with nutrition and disease. Nucleic acids research, 47(D1), D614-D624.

5. Hastings J, Owen G, Dekker A, Ennis M, Kale N, Muthukrishnan V, Turner S, Swainston N, Mendes P, Steinbeck C. (2016)

6. Wishart, D. S., Bartok, B., Oler, E., Liang, K. Y., Budinski, Z., Berjanskii, M., ... & Wilson, M. (2021). MarkerDB: an online database of molecular biomarkers. Nucleic Acids Research, 49(D1), D1259-D1267.

7. Himmelstein, D. S., Lizee, A., Hessler, C., Brueggeman, L., Chen, S. L., Hadley, D., Green, A., Khankhanian, P., & Baranzini, S. E. (2017). Systematic integration of biomedical knowledge prioritizes drugs for repurposing. eLife, 6, e26726.

8. Choi, E., Bahadori, M. T., Song, L., Stewart, W. F., & Sun, J. (2017). GRAM: Graph-based Attention Model for Healthcare Representation Learning. KDD: proceedings. International Conference on Knowledge Discovery & Data Mining, 2017, 787–795.

9. Shen, Z., Zhang, Y. H., Han, K., Nandi, A. K., Honig, B., & Huang, D. S. (2017). miRNA-disease association prediction with collaborative matrix factorization. Complexity, 2017.

10. Fleischmann, A., Darsow, M., Degtyarenko, K., Fleischmann, W., Boyce, S., Axelsen, K. B., ... & Apweiler, R. (2004). IntEnz, the integrated relational enzyme database. Nucleic acids research, 32(suppl_1), D434-D437.

11. Kanehisa, M., & Goto, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. Nucleic acids research, 28(1), 27–30.

12. Golovin, A., Oldfield, T. J., Tate, J. G., Velankar, S., Barton, G. J., Boutselakis, H., ... & Henrick, K. (2004). E-MSD: an integrated data resource for bioinformatics. Nucleic Acids Research, 32(suppl_1), D211-D216.

13. Gaulton, A., Hersey, A., Nowotka, M., Bento, A. P., Chambers, J., Mendez, D., ... & Leach, A. R. (2017). The ChEMBL database in 2017. Nucleic acids research, 45(D1), D945-D954.

14. McKinney, W., & others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).

15. Van Rossum, G., & Drake, F. L. (2009). Python 3 Reference Manual. Scotts Valley, CA: CreateSpace.

16. Neo4j (2012). Neo4j - The World's Leading Graph Database.

17. Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., ... & Taylor, A. (2018, May). Cypher: An evolving query language for property graphs. In Proceedings of the 2018 International Conference on Management of Data (pp. 1433-1445).

18. Wishart, D. S., Tzur, D., Knox, C., Eisner, R., Guo, A. C., Young, N., Cheng, D., Jewell, K., Arndt, D., Sawhney, S., Fung, C., Nikolai, L., Lewis, M., Coutouly, M. A., Forsythe, I., Tang, P., Shrivastava, S., Jeroncic, K., Stothard, P., Amegbey, G., … Querengesser, L. (2007). HMDB: the Human Metabolome Database. Nucleic acids research, 35(Database issue), D521–D526.

19. Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B. A., Thiessen, P. A., Yu, B., Zaslavsky, L., Zhang, J., & Bolton, E. E. (2019). PubChem in 2021: new data content and improved web interfaces. Nucleic Acids Res., 49(D1), D1388–D1395.