Preprint CheMIn Technical Reports ISSN 3061-1075

Digital Representations of Chemical Knowledge

Aleksandar Kondinski [1]

released: 18 September 2026

[1] Institute of Physical and
Theoretical Chemistry
Graz University of Technology
Stremayrgasse 9, 8010 Graz, Austria

Preprint No. 8



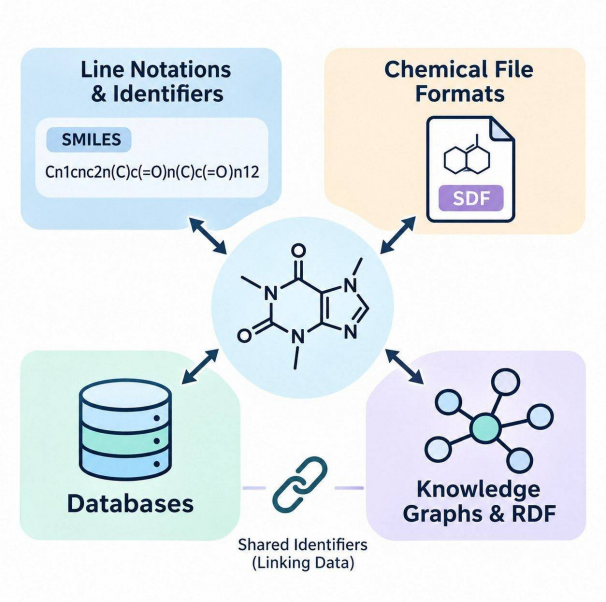
Keywords: chemical information, data formats, databases, knowledge graphs, RDF, SPARQL, chemical identifiers

Edited by
Chemical Modelling and Informatics Group
Institute of Physical and Theoretical Chemistry
Graz University of Technology
Stremayrgasse 9, 8010 Graz
Austria (Europe)

E-Mail: kondinski@tugraz.at
World Wide Web: http://chemin.at/

Abstract

This tutorial provides a knowledge-engineering view of how chemical information is represented in digital form. Following the molecule caffeine as a running example, it surveys line notations and identifiers (SMILES, SELFIES, SMARTS, SMIRKS, InChI, and InChIKey), connection-table and archive formats (MOL, SDF, mol2, CML, CIF, PDB, mmCIF, and FASTA), relational and document databases, and knowledge graphs built with the Resource Description Framework (RDF) and ontologies. A worked federated SPARQL example shows how shared, structure-derived identifiers link compounds, reactions, proteins, genes, and tissue-expression data across public resources. The aim is to help chemists choose representations that keep their data searchable, linkable, and reusable.



Highlights

- Caffeine is followed through digital chemical representations.
- Line notations, record formats, databases, and knowledge graphs.
- Shared identifiers link records across public resources.
- A federated SPARQL query reaches tissue-expression data.

Contents

1 Introduction **3**

2 Line notations and identifiers **4**

2.1 SMILES and canonicalisation 4

2.2 SELFIES, SMARTS, and SMIRKS 5

2.3 InChI and InChIKey 6

3 Records, semantic, and macromolecular formats **7**

3.1 Connection-table and record formats 9

3.2 Semantic and archive formats 9

3.3 Macromolecular formats 10

4 Choosing and combining formats in practice **13**

5 From files to databases **14**

6 Data curation, quality, and bias **16**

7 Knowledge graphs, RDF, and ontologies **18**

7.1 From tables to graphs 18

7.2 RDF triples and Turtle 18

7.3 Ontologies and reasoning 19

8 Querying with SPARQL and federated Linked Data **22**

8.1 Choosing between relational and graph models 22

8.2 SPARQL basics 22

8.3 Identifiers, Linked Data, and chemical resources 23

9 Worked example from a substructure to a tissue **25**

10 Conclusion **27**

References **29**

1 Introduction

A scanned laboratory notebook preserves chemical information, but the contents are not necessarily easy for software to use. Digitisation converts analogue material into digital form. A handwritten spectrum, for example, may become a Portable Document Format (PDF) file. Digitalisation introduces digital tools into existing tasks, such as processing spectra with software or storing measurements in spreadsheets. Digital transformation connects these activities into a broader workflow in which reaction records, molecular identifiers, instrument metadata, and analysis pipelines can be searched, linked, and reused together [8, 9, 32]. How well this works depends strongly on how the underlying chemical information is represented, which is the focus of this tutorial.

A simple example shows why representation matters. Take the number 194.19. On its own, it says very little. Written as 194.19 g mol^{-1} for caffeine, it becomes useful chemical information. If the same record also carries a molecular structure, a standard identifier, and terms from a controlled vocabulary, it can be compared and linked with records held elsewhere. This article takes a knowledge-engineering view of chemistry [37]. From this perspective, the way chemical information is represented determines what software can later search, combine, infer, or analyse using rule-based, statistical, or language-model-assisted methods.

Chemical data formats, databases, and knowledge graphs are often discussed separately. This tutorial brings them together by following one molecule, caffeine, through the different forms in which chemical information can be represented. The path begins with a Simplified Molecular Input Line Entry System (SMILES) string and continues through structure records, databases, and linked knowledge resources. Using the same molecule throughout makes it easier to see what changes from one representation to another and what information may be preserved or lost. The worked example in Sect. 9 uses live public endpoints, and the accompanying code and supplementary materials are archived on Figshare.

Figure 1 gives an overview of the representations covered in the tutorial. Molecular information can be expressed as compact strings, explicit structure records, or semantically enriched archive formats. Feature-based and learned numerical representations used for similarity, prediction, and generation are shown for context but are not discussed further here. Sections 2 to 4 deal with representations of individual molecules, including line notations, identifiers, structure records, and archive formats. Sections 5 to 6 move to collections of molecules and discuss relational and document databases together with data curation. Sections 7 to 9 introduce knowledge graphs, the Resource Description Framework (RDF), ontologies, and the SPARQL Protocol and RDF Query Language (SPARQL). The tutorial closes with a worked query that follows a chemical substructure across several public resources to information on tissue expression.

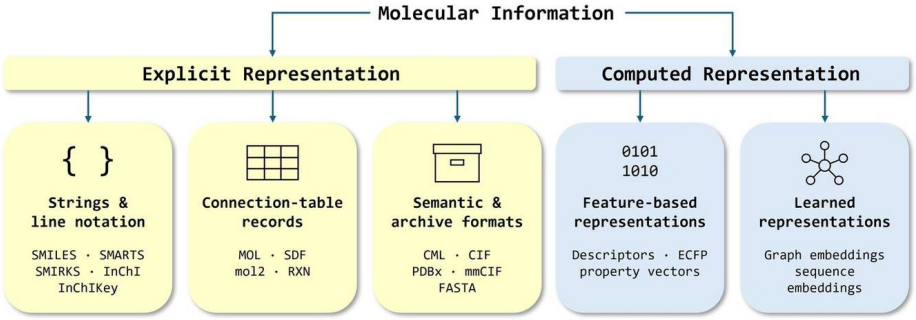


Figure 1: *The landscape of molecular representations. String and line notations encode molecular structure compactly. Connection tables and record formats preserve atoms, bonds, coordinates, and associated data more explicitly. Semantic and archive formats add structured metadata for exchange and deposition. Feature-based and learned numerical representations used for similarity, prediction, and generation are outside the scope of this tutorial.*

2 Line notations and identifiers

Line notations describe chemical structures as text strings. Their main advantage is compactness because a short string can be stored in a database, entered into a search box, copied into a spreadsheet, or passed directly to software. The trade-off is that these strings usually contain less information than coordinate-based structure records and do not carry experimental or descriptive metadata on their own [63]. The formats discussed here serve different purposes. The Simplified Molecular Input Line Entry System (SMILES) and Self-Referencing Embedded Strings (SELFIES) encode molecular structures. SMARTS is a SMILES-based language for describing structural patterns, while SMIRKS uses related syntax to describe formal structural transformations. The International Chemical Identifier (InChI) and InChIKey serve a different purpose and are designed for standardised identification and for matching chemical records across different systems. Keeping these roles separate is important because a notation used to describe or search for a structure is not necessarily suitable as a stable identifier.

2.1 SMILES and canonicalisation

SMILES is a linear notation for describing molecular structure [60]. For caffeine, shown in Fig. 2, one valid SMILES string is `Cn1c(=O)c2c(ncn2C)n(C)c1=O`.

The string describes caffeine's fused xanthine ring system, three N-methyl groups, and two carbonyl groups. SMILES uses a compact set of rules. Atoms are written as element symbols, branches appear in parentheses, and matching numbers mark ring closures. Lower-case symbols denote aromatic atoms, while symbols such as `=` and `#` specify bond types where needed. Stereochemistry can also be encoded using symbols such as `@`, `/`, and `\`. Despite its compact form, a SMILES string contains enough information to reconstruct the molecular graph, which has made the notation widely used for small-molecule

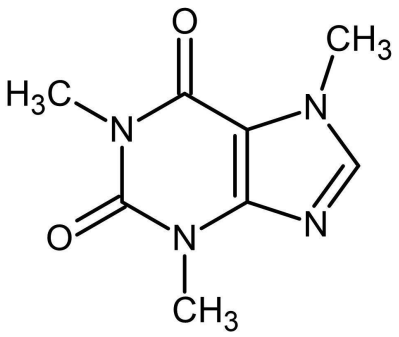


Figure 2: *Two-dimensional structure of caffeine, used as the running molecular example throughout this tutorial.*

exchange, searching, and database records.

A molecule does not have only one possible SMILES string. Different paths through the same molecular graph can produce different strings that describe the same structure. Canonicalisation addresses this by applying a defined atom-ranking and traversal procedure to produce a reproducible form [61]. The result, however, depends on the software used. Two toolkits can therefore generate different canonical SMILES for the same molecule. A canonical SMILES should not be confused with a standardised chemical identifier. It is reproducible within a particular canonicalisation scheme, but it does not guarantee that different databases or software packages will represent a compound with the same string. Weininger's original papers introduced the notation and its canonicalisation approach, while later specifications such as OpenSMILES [52] and the IUPAC SMILES+ initiative [35] have worked towards a more explicit and consistent description of the syntax.

2.2 SELFIES, SMARTS, and SMIRKS

SELFIES was developed with machine-learning applications in mind [39]. Its main aim is robustness. Strings that follow the SELFIES grammar can be decoded into molecular graphs that respect the semantic constraints specified in the grammar, including the bonding capacities of the atoms, which makes the notation useful when molecular strings are generated or modified automatically. This does not mean that every decoded structure is chemically sensible. Formal validity is not the same as chemical plausibility. A decoded structure obeys the encoded valence rules, but this does not guarantee chemical stability or synthetic accessibility, and unusual bonding patterns or strained ring systems can still be produced, and later work has refined both the grammar and the original claims of universal validity [39, 40]. SMILES does not provide the same safeguard, since a small change to a SMILES string can easily make it invalid. Both notations can represent molecular structures, but they were designed with different priorities. SMILES remains common for exchange, storage, and display, while SELFIES is particularly useful in generative molecular design.

SMARTS is used to describe structural patterns rather than individual molecules [63]. A simple SMARTS pattern for a carbonyl group is

```
[#6]=[OX1]
```

Here `[#6]` denotes a carbon atom and `[OX1]` an oxygen atom with one connection. In caffeine, this pattern matches the two carbonyl groups of the xanthine framework. Applied to a molecular database, the same pattern can be used to find other structures containing that motif. SMILES and SMARTS therefore serve different purposes. A SMILES string describes one molecular structure, while a SMARTS pattern defines a structural feature that may occur in many molecules. SMARTS is widely used for substructure searching, filtering, and rule-based pattern matching.

SMIRKS is used to describe structural transformations [23, 63]. It combines SMARTS-like atom definitions with changes in bonding, and atom-map numbers identify corresponding atoms on the two sides of the transformation. For example, `[CH3:1][CH2:2][OH:3]>>[CH3:1][CH:2]=[O:3]` describes the formal conversion of ethanol to acetaldehyde. The mapped atom numbers show which atoms are retained across the transformation. A SMIRKS rule specifies a structural change, but not a reaction mechanism or experimental conditions. This particular example is deliberately specific because the explicit hydrogen counts restrict it to a two-carbon primary alcohol. A more general rule would require broader atom definitions.

2.3 InChI and InChIKey

InChI is a standardised identifier generated algorithmically from a molecular structure [30, 31]. Its main purpose is identification rather than flexible structure notation or searching. InChI provides a consistent way to compare and link chemical records across databases, publications, and software systems. The identifier is organised into layers. These record the molecular formula, atom connectivity, and hydrogen positions, with additional information for charge, stereochemistry, protonation, and isotopic composition when needed. For caffeine, the standard InChI is

```
InChI=1S/C8H10N4O2/c1-10-4-9-6-5(10)7(13)12(3)8(14)11(6)2/h4H,1-3H3
```

The layered structure makes InChI useful for standardised identification, although the resulting string is less convenient than SMILES for reading, drawing, or editing a molecule.

InChIKey is a fixed-length hashed form of InChI that is easier to use for database indexing, web searches, and matching records between systems [31, 54]. The InChIKey for caffeine is

```
RYYVLZVUVIJVGH-UHFFFAOYSA-N
```

An InChIKey contains 27 characters arranged in three parts. The first fourteen characters are derived from the main connectivity layer. The second part contains eight characters derived from the remaining InChI layers, followed by a character indicating whether the InChI is standard or non-standard and a character indicating the InChI version. The final character represents the protonation state. Because the key has a fixed length, it is

convenient for searching and joining records. It is not, however, a replacement for the full InChI because the hashing process is one-way and the molecular structure cannot be reconstructed from the key alone.

InChIKeys can be generated directly from molecular structures using cheminformatics software. With RDKit, for example, the caffeine InChIKey can be obtained with a few lines of Python code.

```
from rdkit import Chem

mol = Chem.MolFromSmiles("Cn1c(=O)c2c(ncn2C)n(C)c1=O")
print(Chem.MolToInchiKey(mol))
# RYYVLZVUVIJVGH-UHFFFAOYSA-N
```

A matching InChIKey assumes that the systems being compared start from equivalent molecular representations. Differences in stereochemistry, protonation state, or structural features that are not normalised equivalently can lead to different InChIs and therefore different InChIKeys. Standard InChI normalises recognised tautomerism, so different tautomeric drawings do not necessarily produce different identifiers [31]. InChIKey is therefore a powerful tool for linking records, but it cannot resolve disagreements in the structure supplied before the identifier is generated.

Table 1 brings these six representations together for comparison. Each serves a different purpose, so the choice depends on what information needs to be represented and how it will be used.

A valid SMILES string is not always a good database join key. The same molecule can be written as different SMILES strings because the molecular graph can be traversed in different ways. For ethanol, both `CCO` and `OCC` describe the same structure. A direct string comparison between two databases may therefore miss a genuine match and make the compound appear to be absent. One solution is to canonicalise the structures with the same toolkit before comparison. A more reliable option for linking records is to use InChI or InChIKey, provided that the underlying structures have been standardised consistently. Otherwise, a failed match may simply reflect a difference in representation rather than a real difference in chemical identity.

3 Records, semantic, and macromolecular formats

Line notations provide a compact way to represent molecular structure, but many chemical workflows need information that cannot be captured in a short string. This may include atomic coordinates, atom-level properties, measured data, or information required for deposition and archiving. The formats discussed in this section address these needs in different ways. Connection-table and record formats such as MOL, Structure-Data File (SDF), and mol2 store atoms and bonds explicitly. Chemical Markup Language (CML) and Crystallographic Information File (CIF) combine structural information with named data fields and metadata. Biomolecular information is commonly stored using

Table 1: *Comparison of line notations and identifier formats, using caffeine unless noted*

Format	Example	Represents	Common use
SMILES	`Cn1c(=O)c2c(ncn2C)n(C)c1=O`	Molecular connectivity in a compact string	Structure exchange, searching, database records
SELFIES	`[C][N][C][=Branch1][C][=O][C][=C]...`	Molecular connectivity using a robust grammar	Generative molecular design
SMARTS	`[#6]=[OX1]`	A structural pattern that can match many molecules	Substructure searching and filtering
SMIRKS	`[CH3:1][CH2:2][OH:3]>>[CH3:1][CH:2]=[O:3]`	A mapped structural transformation	Reaction rules and structure transformations
InChI	`InChI=1S/C8H10N4O2/c1-10-4-9-6-5(10)7(13)12(3)8(14)11(6)2/h4H,1-3H3`	A standardised layered structural identifier	Chemical identification and record linking
InChIKey	`RYYVLZVUVIJVGH-UHFFFAOYSA-N`	A fixed-length hash derived from InChI	Database matching and web searching

the legacy Protein Data Bank (PDB) format, macromolecular Crystallographic Information File (mmCIF), and FASTA. These formats accommodate large molecular structures, biological sequences, and their associated information. Caffeine remains the running example, first as an individual molecule and later as a ligand bound to a protein.

3.1 Connection-table and record formats

The Molecular Design Limited (MDL) CTfile family has been used for chemical structure exchange for decades and remains widely supported by cheminformatics software [15]. A molfile (`.mol`) stores a single molecular structure in a series of defined blocks. These include a three-line header, a counts line giving the number of atoms and bonds, an atom block containing Cartesian coordinates and element symbols, a bond block specifying which atoms are connected and their bond orders, and an `M END` line marking the end of the record. The position of each item within the file carries much of its meaning. For caffeine, a molfile therefore contains an explicit list of its atoms, bonds, and coordinates rather than a compact string. The older V2000 format has several size and representation limits that can become restrictive for larger or more complex structures. V3000 retains the same general purpose while using a more flexible syntax.

The Structure-Data File (SDF) allows several molecular records to be stored in one file together with associated data [15]. Each record contains a molfile followed by named data fields. For caffeine, these could include `<NAME>` with the value Caffeine, `<FORMULA>` with the value C8H10N4O2, and `<INCHIKEY>` with the value RYYVLZVUVIJVGH-UHFFFAOYSA-N. The line `$$$$` marks the end of a record. SDF is therefore convenient for compound collections in which structures are stored together with properties, assay results, or other annotations. One limitation is that field names and conventions are not standardised between datasets. Two valid SDF files may use different names, units, or missing-value conventions for the same kind of information.

Mol2 was developed for molecular modelling and places more emphasis on information needed by modelling software. A mol2 record is divided into tagged sections such as `@<TRIPOS>MOLECULE`, `@<TRIPOS>ATOM`, and `@<TRIPOS>BOND` [63]. Atoms can be assigned modelling-oriented types such as `C.2`, `N.am`, or `O.2`, and partial charges may also be included. These features have made mol2 useful in docking and force-field workflows. They also introduce a source of inconsistency because atom types, aromaticity, and partial charges may be assigned differently by different programs. A mol2 file should therefore be interpreted together with the conventions of the software that produced it.

3.2 Semantic and archive formats

Chemical Markup Language (CML) represents chemical information using named Extensible Markup Language (XML) elements rather than relying on fixed positions or columns [22, 48, 49]. A CML record can contain a `<molecule>` element together with information such as the compound name, formula, and identifiers. For caffeine, an InChIKey could be written as `<identifierconvention="InChIKey">RYYVLZVUVIJVGH-UHFFFAOYSA-N</identifier>`. The molecular structure itself is commonly divided into an `<atomA`

rray> and a <bondArray>, with atoms referred to by identifiers. Because the meaning of each item is given explicitly by the element names, CML records can be checked, transformed, and incorporated into larger XML-based systems. CML is useful when chemical information needs to be expressed in a structured and machine-readable form, although it is less common than SMILES or SDF for routine small-molecule exchange.

The Crystallographic Information File (CIF) is widely used for the exchange and archiving of crystallographic data [11, 24]. A CIF record is organised through named data items. A `data_...` block begins the record, individual quantities are written as tag and value pairs, and repeated information is placed in `loop_` tables. The format can store unit-cell parameters, space-group information, experimental details, uncertainties, and atomic positions given as fractional coordinates. Caffeine provides a useful real example. Entry 2100202 in the Crystallography Open Database is a deposited caffeine structure with the formula C8 H10 N4 O2. The deposited formula weight is 194.20 g mol^{-1}, while the value calculated using current standard atomic weights is 194.19 g mol^{-1}. This small difference illustrates how even well-structured records can contain minor differences arising from rounding or reporting conventions. A short excerpt from the record is shown below.

```
data_2100202
_chemical_formula_moiety          'C8 H10 N4 O2'
_chemical_formula_sum             'C8 H10 N4 O2'
_chemical_formula_weight          194.20
_chemical_name_common             caffeine

loop_
_publ_author_name
'Derollez, Patrick'
'Correia, Natalia T.'
'Danede, Florence'
'Capet, Frederic'
'Affouard, Frederic'
'Lefebvre, Jacques'
'Descamps, Marc'
```

The excerpt shows the two main organising features of CIF. Single values are given as named data items, while repeated values are grouped in a `loop_` block. CIF is not intended as a general molecular coordinate format. Its purpose is to preserve a crystallographic structure together with the information needed to interpret, assess, and archive it.

3.3 Macromolecular formats

Macromolecular structures bring a different set of practical problems. A single record may contain thousands of atoms together with residues, chains, ligands, and extensive archive information. To keep caffeine as the running example, this section uses entry 3RFM from the Protein Data Bank, which contains the human adenosine A2A receptor

in complex with caffeine [5]. Caffeine therefore reappears here as a ligand bound to a macromolecular structure.

The legacy PDB format uses fixed-width records. Each line begins with a label such as `HEADER`, `TITLE`, or `ATOM`, while fixed column positions specify information such as atom name, residue name, chain identifier, residue number, coordinates, occupancy, temperature factor, and element symbol. This format was central to the early development of structural biology, but its reliance on fixed column widths becomes restrictive as structures and annotations become larger and more complex.

The macromolecular Crystallographic Information File (mmCIF) is the standard archival format used by the Protein Data Bank. Like CIF, it uses named data items and looped tables instead of fixed-width columns [11, 63]. The excerpt below shows three protein atoms from the caffeine binding site in PDB entry 3RFM.

```
data_3RFM
loop_
_atom_site.group_PDB
_atom_site.id
_atom_site.type_symbol
_atom_site.label_atom_id
_atom_site.label_comp_id
_atom_site.label_asym_id
_atom_site.label_seq_id
_atom_site.Cartn_x
_atom_site.Cartn_y
_atom_site.Cartn_z
ATOM 443 N N  ILE A 66  6.982 -41.217 -38.881
ATOM 444 C CA ILE A 66  5.968 -40.888 -37.886
ATOM 445 C C  ILE A 66  4.644 -40.603 -38.566
```

Unlike the legacy PDB format, mmCIF does not rely on fixed field widths. Each value is identified by a tag, and new data items can be added without altering the basic structure of the record. This flexibility makes mmCIF better suited to large and extensively annotated macromolecular structures and led to its adoption as the standard format for deposition.

FASTA represents a different kind of macromolecular information. It originated with the sequence-analysis work of Pearson and Lipman [43, 53] and stores biological sequences rather than three-dimensional structures. A FASTA record begins with a header line marked by >, followed by the sequence written in one-letter residue codes. For chain A of 3RFM, the header can be written as `>3RFM_1|ChainA|AdenosinereceptorA2a|Homosapiens(9606)`. FASTA preserves sequence order but contains no atomic coordinates, bonding information, or crystallographic metadata. It is included here because a biological sequence is itself an important molecular representation and is widely used for sequence databases, alignment, and similarity searching.

Table 2 summarises the main differences between these formats. The comparison focuses on how each record is organised, where it is commonly used, and what information or flexibility it may lack.

Table 2: *Comparison of record, semantic, and macromolecular formats*

Format	Record organisation	Common use	Main limitation
MOL	Fixed header, counts, atom, and bond blocks	Single molecular structures	V2000 becomes restrictive for larger or more complex records
SDF	Molfile plus named data fields, with `$$$$` separating records	Compound collections and assay data	Field names, units, and missing-value conventions vary
mol2	Tagged `@<TRIPOS>` sections with atom types and charges	Docking and molecular modelling	Atom typing and charge conventions differ between programs
CML	XML elements with explicit names and identifiers	Structured chemical data exchange	Less widely used for routine small-molecule exchange
CIF	Named data items and `loop_` tables	Crystallographic deposition and archiving	Designed specifically for crystallographic data
PDB	Fixed-column records	Legacy macromolecular structures	Limited by fixed field widths and record conventions
mmCIF	Named tags and looped tables	Modern macromolecular archiving	More verbose than the legacy PDB format
FASTA	Header line followed by a one-letter sequence	Sequence databases and alignment	No three-dimensional structure or archive metadata

Seen together, these formats show how the same chemical entity can be represented for different purposes. Caffeine can be written as a SMILES string for compact exchange, identified by InChI and InChIKey, stored as an explicit atom and bond record in a molfile, and combined with additional properties in SDF. CML expresses similar information through named elements, while the deposited CIF record COD 2100202 places caffeine in a crystallographic context. In PDB entry 3RFM, caffeine appears as a ligand bound to a protein, with the same complex also available in mmCIF. Each representation serves a different purpose and preserves a different set of information. Moving between formats therefore requires some care because coordinates, metadata, naming conventions, or other details may be changed or lost during conversion.

A file can be syntactically valid and still be difficult to combine with another dataset. One SDF collection might record caffeine with `<MELTING_POINT_C>` = 235, while another uses `<mp>` = 508 without stating that the value is in kelvin. Both files can be read correctly, but merging them without checking field names and units would produce misleading results. Missing values cause similar problems when one dataset leaves a field empty and another uses `NA`. Before combining datasets, field names, units, and missing-value conventions therefore need to be checked and, where necessary, standardised. This is discussed further in Sect. 6.

4 Choosing and combining formats in practice

Sections 2 and 3 introduced the main formats and what they preserve. In practice, however, a chemical workflow rarely uses only one of them. Caffeine might be written as SMILES for compact exchange, converted to InChI or InChIKey for matching records across systems, stored as a molfile when explicit atoms and bonds are needed, and exported as SDF when properties must remain attached to the structure. CIF, PDB, or mmCIF may be used when the structure becomes part of an archive. Moving between these formats can also remove information. Converting a three-dimensional molfile to SMILES discards the atomic coordinates. A new three-dimensional structure generated later may be reasonable, but it will not reproduce the original coordinates. Converting SDF data to a spreadsheet may also lose information about how fields were defined. Even canonicalising a SMILES string with another toolkit can change the string while leaving the molecular structure unchanged. Choosing a format therefore also means deciding which information must be retained.

JavaScript Object Notation (JSON) appears in many of these workflows even though it is not a chemical format. It is a general-purpose format for structured data. Web services such as the PubChem PUG-REST Application Programming Interface (API) can return compound records as JSON, and many computational datasets are distributed as JSON or JSON Lines. JSON itself does not define chemical meaning. The meaning comes from the field names and conventions chosen by the application that produces the record. It is therefore useful as a way of transporting structured chemical data, but it does not replace formats such as SMILES, SDF, or CIF. JSON also appears again in Sect. 5 when document-oriented databases are discussed.

Conversions between chemical formats are usually handled by software rather than per-

formed by hand. Open-source toolkits such as RDKit [42] and Open Babel [51] can read and write many of the formats introduced above. They also provide functions for tasks such as canonicalisation, structure depiction, and identifier generation. The Chemistry Development Kit (CDK) [57] provides similar functionality in Java and is used in a number of chemical information systems. These tools make conversion straightforward, but they cannot decide which information a particular workflow can afford to lose. That decision still belongs to the user.

Problems often appear when records from different sources need to be combined. One database may provide caffeine only as a SMILES string, while another may provide an SDF record identified only by a local database number. Without a shared identifier, the two records cannot be linked directly. Their structures first have to be standardised and a common identifier generated from each of them. Adding InChI or InChIKey when records are created or exported makes this later comparison much easier. A format choice that seems unimportant when one dataset is considered alone can therefore become important when that dataset has to be connected with another.

5 From files to databases

File formats are useful for storing individual molecules, but they become difficult to manage when a collection grows. A laboratory may eventually need to search thousands of compounds, connect them with measurements and spectra, update records, and keep track of where the information came from. Chemists have dealt with this problem for a long time. Organised chemical abstracting began in the nineteenth century with the *Pharmaceutisches Centralblatt*, later known as the *Chemisches Zentralblatt* [12]. *Chemical Abstracts* followed in 1907. In 1965, the Chemical Abstracts Service (CAS) introduced its Chemical Registry System, which assigned stable identifiers to chemical substances [25, 62]. Resources such as the CAS Registry, Beilstein, Reaxys, and SciFinder later showed the value of organising chemical information systematically. They also showed that building such resources requires sustained editorial effort to collect, check, and standardise chemical information.

Many laboratories still begin with flat files such as Comma-Separated Values (CSV) tables, SDF collections, and JSON records. These are convenient at small scale, but problems appear as the collection grows [18]. The same compound may be entered several times under different names. Measurements may be repeated in different files. Answering a new question may require another script that reads and combines several datasets. Problems can also arise when several users or programs update related files at the same time. A database management system provides a more controlled way to store the records, enforce consistency, manage simultaneous changes, and recover data when something goes wrong.

A relational database stores information in tables [18]. Separate tables may contain compounds, measurements, and samples. Each row represents a record and each column an attribute. The database schema defines how these tables are organised, including their columns, data types, and relationships. A primary key uniquely identifies a record, while a foreign key links it to a record in another table. Caffeine can therefore be stored once

in a compound table, with measurements, spectra, and references linked back to the same entry. Figure 3 gives a simple chemical example of this relational structure.

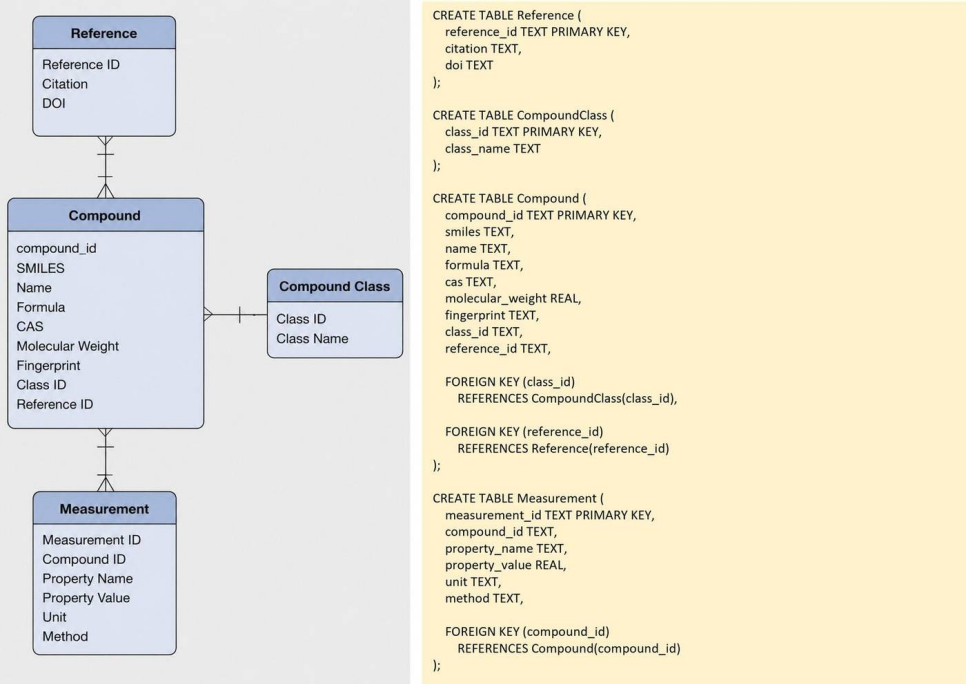


Figure 3: *A simple chemistry database shown first as a conceptual schema and then as a relational implementation using tables, primary keys, and foreign keys.*

Structured Query Language (SQL) is commonly used to retrieve information from relational databases [18]. Instead of reading each file directly, the user asks the database for the records that satisfy a particular condition. The following query joins compounds with their measured properties and returns records for the molecular formula of caffeine.

```
SELECT c.name, m.property_name, m.property_value, m.unit
FROM Compound c
JOIN Measurement m
    ON c.compound_id = m.compound_id
WHERE c.formula = 'C8H10N4O2';
```

Document-oriented databases store information differently. Instead of separating compounds and measurements across several tables, related information can be kept together in a single nested record [47]. This can be convenient when records vary in structure, for example for instrument output, spectra, or simulation results. A caffeine record might look like this.

```
{
  "compound_id": "C001",
  "name": "Caffeine",
  "formula": "C8H10N4O2",
```

```
  "smiles": "Cn1c(=O)c2c(ncn2C)n(C)c1=O",
  "measurements": [
    { "property_name": "molecular_weight", "property_value": 194.19, "unit": "g/mol" },
    { "property_name": "melting_point", "property_value": 235, "unit": "C" }
  ]
}
```

Databases also use indexes to make searching faster. Ordinary indexes can locate records through identifiers such as a compound ID, registry number, or molecular formula. Structure searching needs more specialised methods. Molecular fingerprints are often used to screen a large collection before a more detailed substructure or similarity comparison is carried out [13, 56]. Relational database systems also support transactions and recovery, which become important when several users or programs modify the same collection [18].

Tables work very well for many chemical datasets, but some questions are easier to express as relationships. A chemist may want to follow all precursors leading to a compound, trace a reaction network, or follow the provenance of a result through several intermediate records. SQL can express such recursive queries, but graph-based models make relationships more explicit [34]. Graph databases such as Neo4j store entities and their connections directly and can be queried with languages such as Cypher. The Resource Description Framework (RDF) stores information as linked triples and is queried using SPARQL. These ideas are introduced in Sect. 7. The difference between a database and a knowledge base is therefore useful here. A database may store caffeine with its name, formula, and identifier. A knowledge base can also connect caffeine to chemical classes, reactions, proteins, publications, and controlled vocabulary terms [34, 47].

Moving information into a database does not automatically remove problems in the underlying data. Caffeine could still be entered twice, once as “Caffeine” and once as “1,3,7-trimethylxanthine”, if no shared identifier connects the two records. A search based only on names may then miss some measurements or count the compound twice. Database constraints and uniqueness rules can help, but they depend on good identifiers and careful curation when records are added. These issues are considered in the next section.

6 Data curation, quality, and bias

Databases and knowledge graphs are only as useful as the data they contain. Data curation is the process of selecting, cleaning, standardising, documenting, and maintaining chemical data so that they remain useful over time. This can involve resolving duplicate records, standardising identifiers and units, correcting or flagging errors, recording provenance, and documenting how records were selected. A curated dataset therefore differs from a raw collection because these decisions have been made explicitly and can be traced.

At larger scale, much of this work is automated. Standardisation pipelines such as the ChEMBL Structure Pipeline [4], MolVS, and routines available in Open Babel apply a defined sequence of operations to incoming structures. These may include removing salts or solvents, selecting a parent structure, normalising tautomers and charge states,

checking aromaticity and valence, and flagging records that fail basic validity checks. Deduplication follows the same principle discussed earlier for identifiers. Two records with different names or SMILES strings may still represent the same compound, and standardised identifiers such as InChIKey can help detect this once the structures have been processed consistently.

Data quality can be considered in several ways. Accuracy asks whether a value is correct. Completeness asks whether the expected records and fields are present. Consistency asks whether the same entity is represented in the same way across a collection. Provenance asks whether a value can be traced back to its source. These questions arise frequently in chemical datasets. A compound name may refer to more than one structure. A solubility value may be reported without temperature or solvent. The same measurement may appear in different databases with different units. Such problems are common enough that they should be expected when data from several sources are combined.

Caffeine can be used to illustrate how these issues appear in practice. Caffeine is often handled as the monohydrate, $C_8H_{10}N_4O_2 \cdot H_2O$, with a molar mass of about 212.2 g mol^{-1} rather than 194.19 g mol^{-1}. If one supplier record lists the hydrate and another the anhydrous compound, the two records describe the same parent structure but different substances. Their InChIKeys differ unless the water is removed during standardisation, so the records are not matched, and a molar mass or a mass-based concentration calculated from the wrong form is off by about 9%. A standardisation pipeline that removes the water and keeps the caffeine parent structure makes the two records comparable, but the original form should still be recorded, because a measurement made on the hydrate describes the hydrate. Parent-structure selection works in the same way for salts, which are reduced to their largest organic component. Whatever is discarded during this step should remain traceable. A caffeine solubility value shows a different problem. Reported in water as roughly 20 g L^{-1} at room temperature and several times higher at 80 °C, caffeine appears in different sources with values that look contradictory. Without the temperature, the values cannot be compared, and an automated check may wrongly flag one of them as an error or merge them as duplicates. Recording the temperature, solvent, and source with each value removes this ambiguity and illustrates why provenance is part of data quality.

Curated datasets can also contain bias. Some classes of compounds are studied far more often than others. Successful experiments are more likely to be reported than unsuccessful ones, and properties that are easy to measure are often represented more densely than those that are difficult or expensive to obtain. These imbalances matter when the data are used to build predictive models. A solubility model trained mainly on drug-like molecules, for example, should not automatically be expected to perform equally well for ionic liquids or metal complexes. The range of chemistry represented in the training data therefore helps define the applicability domain of the model and the region in which its predictions can be interpreted with confidence.

7 Knowledge graphs, RDF, and ontologies

7.1 From tables to graphs

Relational databases and knowledge graphs organise information differently. A relational database stores records in tables and links them through keys. A knowledge graph represents the entities and their relationships directly. Compounds, reactions, proteins, instruments, and publications can therefore be connected as nodes and labelled edges [34]. This representation is well suited to questions that move across several types of information, for example from a compound to a reaction, then to the protein that catalyses it, the corresponding gene, and the tissues in which that gene is expressed.

Knowledge graphs are closely connected to the idea of the Semantic Web, in which information on the Web is given enough structure and meaning to be processed and connected by software [6]. Simply placing data online is not enough. The data should be machine-readable, identified using Uniform Resource Identifiers (URIs), and linked to related resources. The 5-star Open Data model describes this progression from data that are merely available online to data that are structured, use open formats, have URI-based identifiers, and are linked to other datasets [6, 7]. This approach is already used in research and industry. AstraZeneca's Biological Insights Knowledge Graph, for example, brings together information on targets, diseases, clinical evidence, assays, publications, and internal research data in one connected resource for drug discovery [21]. The value of the graph lies in being able to follow these relationships across information that would otherwise sit in separate databases.

7.2 RDF triples and Turtle

Linked Data relies on a small set of Web technologies [7, 14, 26]. Uniform Resource Identifiers (URIs) give resources stable names, while the Hypertext Transfer Protocol (HTTP) provides a way to access them on the Web. The Resource Description Framework (RDF) provides the data model, and SPARQL provides the query language discussed in Sect. 8. Together, these technologies allow one dataset to refer directly to another. A compound in one resource can therefore be linked to the same compound elsewhere, to a biochemical reaction, or to a protein in UniProt.

RDF represents information as triples consisting of a subject, a predicate, and an object [14]. The subject is the entity being described. The predicate states the relationship or property, and the object is either another entity or a literal value.

```
<http://example.org/chem/M001>
    <http://example.org/chem/hasFormula>
    "C8H10N4O2" .
```

A collection of RDF triples forms a directed, labelled graph in which entities are nodes and predicates are the links between them. In the example above, `C8H10N4O2` is not

simply stored as a text value. The predicate `hasFormula` states explicitly that it is the molecular formula of the chemical entity.

Turtle is a compact and human-readable syntax for writing RDF [55]. Prefixes shorten repeated URI fragments, and several statements about the same subject can be grouped together.

```
@prefix ex: <http://example.org/chem/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ex:M001 ex:hasName "caffeine" ;
        ex:hasFormula "C8H10N4O2" ;
        ex:hasMolarMass "194.19"^^xsd:decimal ;
        ex:hasRole ex:stimulant .
```

The expression `"194.19"^^xsd:decimal` identifies 194.19 as a numerical value rather than ordinary text, which allows numerical comparisons in queries. The term `ex:stimulant` is only a local example. If every dataset creates its own terms for concepts such as molecular formula, molar mass, or biological role, those datasets become difficult to combine. Shared vocabularies and ontologies help avoid this problem. Resources such as ChEBI and CHEMINF provide established chemical terms and relationships that can be reused across datasets [16, 28, 29]. A real knowledge graph could therefore use a ChEBI role such as CHEBI:35337 for a central nervous system stimulant instead of creating a new local term.

Basic RDF triples also have limits. A triple can state a fact, but experimental data often need additional information about where that fact came from, who reported it, or under which conditions it was measured. Named graphs provide one way to associate provenance with groups of triples. RDF-star and SPARQL-star provide another by allowing statements about individual triples [27]. This becomes particularly useful in chemical knowledge graphs that contain measurements and experimental observations as well as structural information.

7.3 Ontologies and reasoning

An ontology gives formal meaning to the terms used in a knowledge graph by defining classes, properties, hierarchies, and constraints [33, 34]. In chemistry, this helps distinguish a molecular entity from a chemical role, a structural descriptor from an experimental property, or a spectral feature from an instrument setting. Services such as the Ontology Lookup Service allow researchers to search existing ontologies and reuse established terms instead of creating a new vocabulary for every dataset [45].

Ontology-based knowledge graphs are often described using two components. The terminological component, known as the TBox, contains classes, subclasses, properties, and logical restrictions. The assertional component, known as the ABox, contains statements about individual entities. For illustration, an ontology might state in its TBox that every xanthine derivative belongs to a broader alkaloid class. Its ABox might then state that

caffeine is a xanthine derivative and has the molecular formula C8H10N4O2. A triplestore can store and query both kinds of information [14, 26, 34]. A simplified example is shown below.

```
# TBox
ex:XanthineDerivative rdfs:subClassOf ex:Alkaloid .

# ABox
ex:Caffeine a ex:XanthineDerivative ;
            ex:hasFormula "C8H10N4O2" .
```

The statement `ex:Caffeineaex:Alkaloid` is not written explicitly. It is inferred from the class hierarchy. In this illustrative ontology, `ex:XanthineDerivative` is defined as a subclass of `ex:Alkaloid`, and caffeine is assigned to the former class. A reasoner can therefore also classify caffeine as an alkaloid. This kind of inference is useful when chemical or biological classifications are organised hierarchically, since broader classifications do not need to be entered separately for every entity.

Whether an inferred statement can be retrieved depends on how the knowledge graph is configured. Many public SPARQL endpoints query the triples that have been stored explicitly and do not perform full ontology reasoning for every request. An inferred statement can still appear in the results if the inference has already been materialised. In that case, a reasoner calculates the additional statements in advance and stores them as ordinary triples. A query that returns no result may therefore reflect the configuration of the endpoint rather than a problem with the ontology itself [33].

RDF Schema (RDFS) provides basic mechanisms such as subclass and subproperty relationships [10]. The Web Ontology Language (OWL) extends this with richer modelling options, including equivalent classes, property restrictions, cardinality constraints, and disjoint classes [33]. Ontologies can be created interactively with tools such as Protégé [50] or maintained programmatically with libraries such as OWLReady2 [41]. The latter is especially useful when ontology development forms part of a reproducible workflow. Ontologies and knowledge graphs are also being combined with software agents in chemical and materials research [1, 37]. Figure 4 presents an example in which concrete molecular entities and more abstract chemical concepts are represented within the same knowledge model.

Reasoning is only as reliable as the statements on which it depends. If an ontology defines two classes as equivalent when they are only chemically similar, a reasoner will treat every member of one class as a member of the other. The resulting inference may be logically correct according to the ontology while still being chemically wrong. The same problem can arise from an incorrect subclass relationship. Equivalence and subclass statements should therefore be treated as strong modelling decisions. Their consequences should be checked against chemical knowledge before inferred results are accepted.

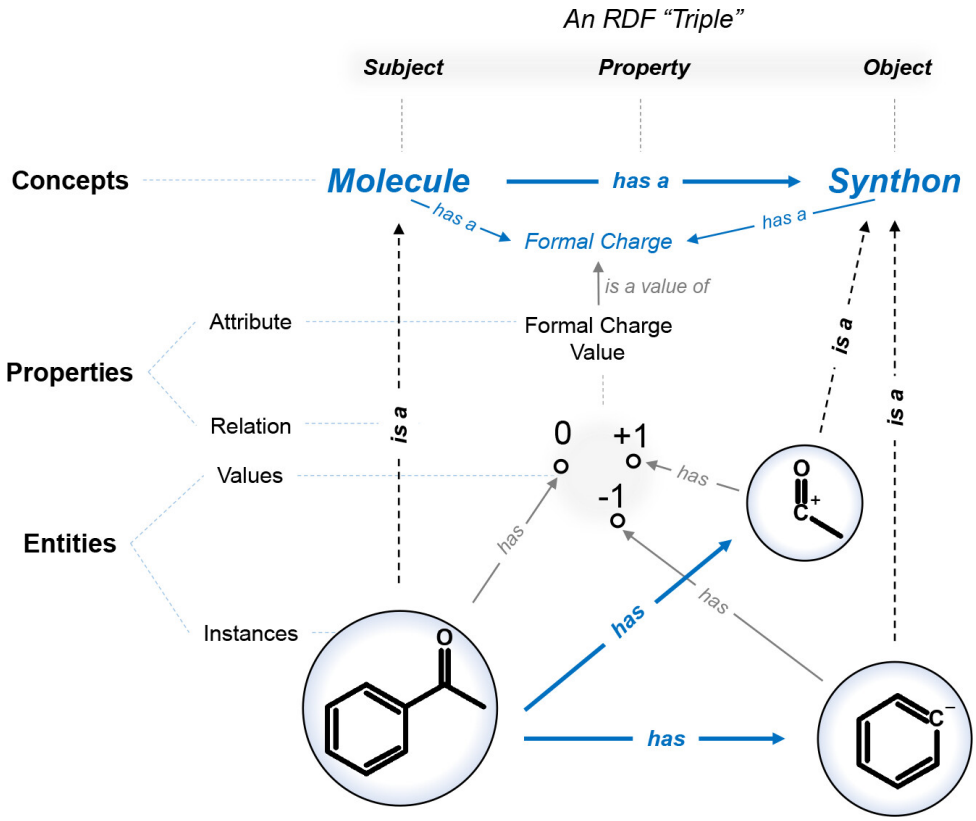


Figure 4: *Concrete molecular entities and abstract synthetic concepts represented in one knowledge model. The upper part shows the RDF pattern of subject, property, and object. The lower part shows an ontology structure containing abstract concepts, properties and values, and concrete entities such as molecules and synthons. Adapted from Ref. [37] under the Creative Commons Attribution 4.0 International (CC BY 4.0) licence.*

8 Querying with SPARQL and federated Linked Data

8.1 Choosing between relational and graph models

Relational databases and knowledge graphs suit different kinds of problems [18, 34, 47]. Relational databases work well when the data structure is stable, records fit naturally into tables, and transaction safety matters. Knowledge graphs are better suited to heterogeneous data in which relationships between entities are central. Chemistry often uses both. Assay measurements and instrument records fit naturally into tables, while connections between compounds, reactions, proteins, genes, publications, and ontology terms are easier to follow in a graph. The two models can also be used together. Figure 5 compares a caffeine solubility observation represented in a relational database and as an RDF graph.

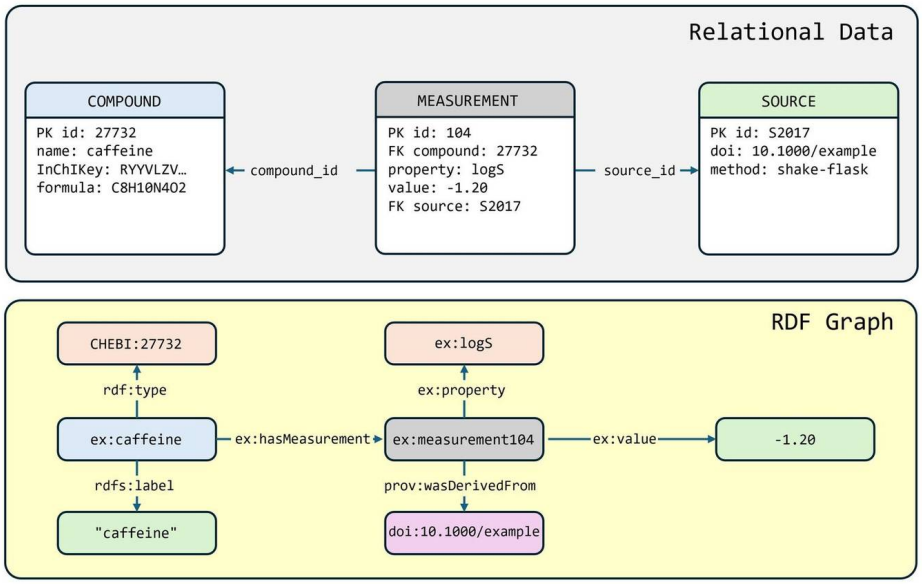


Figure 5: *The same caffeine solubility observation represented using two data models. In the relational model, compound, measurement, and source records are connected through foreign keys. In the RDF graph, the same information is represented as linked entities and properties, so the measurement and its provenance can be followed directly.*

8.2 SPARQL basics

SPARQL is the query language developed by the World Wide Web Consortium (W3C) for RDF data [26]. Its basic form is similar to the SQL example in Sect. 5. A `SELECT` clause specifies which variables should be returned, while a `WHERE` clause describes the pattern that should be matched. In SPARQL, the pattern is made of RDF triples rather than rows in tables. Variables begin with a question mark.

```
PREFIX ex: <http://example.org/chem/>
```

```
SELECT ?formula
WHERE {
  ?compound ex:hasName "caffeine" .
  ?compound ex:hasFormula ?formula .
}
```

This query asks for the molecular formula of an entity whose name is caffeine. SPARQL also provides several features that are useful when data are incomplete or come from different sources. `FILTER` can restrict results using numerical or textual conditions. `OPTIONAL` allows a result to remain even when an additional value is missing. `UNION` allows alternative graph patterns, while `CONSTRUCT` can return a new RDF graph instead of a table of results [26].

SPARQL can also retrieve data from a remote endpoint. The `SERVICE` keyword sends part of the query to another SPARQL service, so results from separate resources can be combined. The example below links a local compound record to Wikidata through its InChIKey. Wikidata represents InChIKey using the property `wdt:P235` [26, 59].

```
PREFIX ex:   <http://example.org/chem/>
PREFIX wdt:  <http://www.wikidata.org/prop/direct/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?localCompound ?wikidataCompound ?label
WHERE {
  ?localCompound ex:hasInChIKey ?inchikey .

  SERVICE <https://query.wikidata.org/sparql> {
    ?wikidataCompound wdt:P235 ?inchikey ;
                      rdfs:label ?label .
    FILTER(LANG(?label) = "en")
  }
}
```

The two resources are linked through the InChIKey. The local database and Wikidata may use different internal identifiers and data models, but records can still be matched when the same standardised molecular structure gives the same InChIKey. Here, the identifier introduced in Sect. 2.3 becomes the link between separate information systems.

8.3 Identifiers, Linked Data, and chemical resources

Linked Data depends on identifiers that are stable, unique, and resolvable. Berners-Lee's principles propose that resources should be identified using Hypertext Transfer Protocol (HTTP) Uniform Resource Identifiers (URIs), that these identifiers should return useful information when accessed, and that the returned information should contain links to other resources [6, 7]. In chemistry, this means that compounds, reactions, proteins, and ontology terms should be identified in ways that allow different systems to recognise the same

entity. InChI and InChIKey are useful for this because they are generated from molecular structure. When two resources start from equivalent standardised structures, the resulting InChIKey can provide a common link between their records [31]. Caffeine, for example, has identifiers in PubChem, ChEBI, ChEMBL, and Wikidata. Table 3 lists examples from each resource. Matching these records through a common structure-derived identifier helps establish that they refer to the same chemical entity [31, 59, 64].

Table 3: *Identifiers and URIs for caffeine in major chemical information resources*

Resource	Identifier	Example URI
PubChem	Compound ID (CID)	`http://rdf.ncbi.nlm.nih.gov/pubchem/compound/CID2519`
ChEBI	ChEBI ID	`http://purl.obolibrary.org/obo/CHEBI_27732`
ChEMBL	ChEMBL ID	`https://www.ebi.ac.uk/chembl/compound/inspect/CHEMBL113`
Wikidata	QID	`http://www.wikidata.org/entity/Q60235`

A stable identifier allows a record to be linked to other information. A compound may connect to an ontology class, a biochemical reaction, a protein target, a publication, or a dataset. This supports the FAIR principles of making data Findable, Accessible, Interoperable, and Reusable [64]. Shared identifiers are especially important for interoperability because they let different resources refer to the same entity even when their internal database structures differ.

Several chemical and biological resources already provide data in forms that support this kind of linking. PubChemRDF represents PubChem substance and compound records using the Resource Description Framework (RDF) and makes use of vocabularies and ontologies including CHEMINF, the Semanticscience Integrated Ontology (SIO), and ChEBI [19]. PubChem distributes RDF data for bulk use and third-party services can provide SPARQL access to these data [20]. ChEBI provides chemical entities of biological interest together with ontology information in formats including Open Biomedical Ontologies (OBO), the Web Ontology Language (OWL), and RDF [16, 29]. Its main access routes include the web interface and Representational State Transfer (REST) services, while third-party infrastructure such as IDSM provides SPARQL access and chemical structure searching [20, 38]. ChEMBL connects bioactive compounds with assays, targets, documents, and reported activities, and its data are also available in RDF form [46]. Wikidata provides a public knowledge graph that can connect chemical entities with a much broader range of scientific, bibliographic, and general information [59].

Three further resources are used in the worked example in Sect. 9. Rhea is a curated knowledge base of biochemical reactions in which reactions have stable identifiers and are linked to participating ChEBI compounds [2, 44]. UniProt provides information on proteins, including their functions, sequences, and links to genes and pathways [58]. Bgee

provides gene-expression information across species, tissues, and developmental stages [3]. Together, these resources make it possible to begin with a chemical structure and follow linked identifiers through reactions, proteins, genes, and tissue-expression data.

9 Worked example from a substructure to a tissue

The concepts introduced above come together in a worked example that starts from a molecular substructure and ends with tissue-expression information for a related gene. The workflow moves across several independently maintained public resources, with shared identifiers linking one step to the next. Figure 6 outlines the five steps.

1. A substructure search against ChEBI is carried out through the IDSM chemical structure search service to find compounds containing the selected pattern [20, 38].
2. Rhea is queried for biochemical reactions in which one selected compound participates [2, 44].
3. UniProt is queried for reviewed human proteins that catalyse one selected reaction [58].
4. UniProt is then used to map one selected protein to its Ensembl gene [17, 58].
5. Bgee is queried for tissues in which that gene is expressed [3].

The substructure used in the first step is the xanthine, or purine-2,6-dione, ring system found in caffeine. Running the search returns 186 ChEBI compounds containing this core, including caffeine itself. Caffeine is a useful first choice because it also shows that a valid query does not always return the type of result being sought. Rhea contains reactions involving caffeine methylation and demethylation, but the enzymes annotated for these reactions are from plants and bacteria rather than humans. The query for reviewed human proteins therefore returns no result for the six caffeine-related Rhea reactions considered here. The workflow can then continue with xanthine itself. Its oxidation to urate is recorded as Rhea reaction 16669 and is catalysed in humans by xanthine dehydrogenase or oxidase, UniProt entry P47989 and gene *XDH*. UniProt maps this protein to Ensembl gene ENSG00000158125. Bgee then reports expression of this gene in many tissues, including kidney, liver, and intestine.

The workflow uses four public SPARQL endpoints. UniProt is used twice, first to identify proteins and then to obtain the corresponding gene information.

```
IDSM_ENDPOINT    = "https://idsm.elixir-czech.cz/sparql/endpoint/chebi"
RHEA_ENDPOINT    = "https://sparql.rhea-db.org/sparql"
UNIPROT_ENDPOINT = "https://sparql.uniprot.org/sparql"
BGEE_ENDPOINT    = "https://www.bgee.org/sparql/"
```

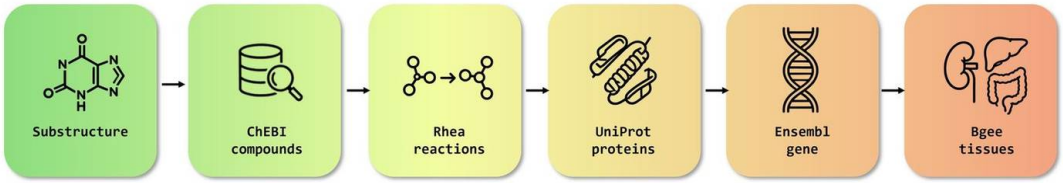


Figure 6: *Overview of the federated SPARQL workflow. The search begins with a chemical substructure and then follows linked identifiers through ChEBI compounds, Rhea reactions, UniProt proteins, Ensembl genes, and Bgee tissue-expression data.*

Each step can be carried out as a standard HTTP request containing a SPARQL query. The response is returned as JSON and the identifier needed for the next step can then be extracted. No specialised SPARQL library is required. The same general pattern can be used for each endpoint.

```
import requests

query = "SELECT ... WHERE { ... }"
response = requests.get(
    RHEA_ENDPOINT,
    params={"query": query, "format": "json"},
    headers={"Accept": "application/sparql-results+json"},
)
rows = response.json()["results"]["bindings"]
```

As an example, the Rhea step is shown here as a complete query. It asks for all Rhea reactions in which xanthine (ChEBI identifier CHEBI:17712) takes part on either side of the equation.

```
PREFIX rh:    <http://rdf.rhea-db.org/>
PREFIX CHEBI: <http://purl.obolibrary.org/obo/CHEBI_>

SELECT DISTINCT ?reaction ?accession ?equation
WHERE {
  ?reaction rh:accession ?accession ;
            rh:equation  ?equation ;
            rh:side/rh:contains/rh:compound/rh:chebi CHEBI:17712 .
}
ORDER BY ?accession
```

The two `PREFIX` lines shorten the Rhea and ChEBI URIs. The property path `rh:side/rh:contains/rh:compound/rh:chebi` leads from a reaction through one of its two sides and a participant to the ChEBI compound, so that no intermediate variables need to be named. `DISTINCT` avoids repeating a reaction in which xanthine appears on both sides. When this query was run against the Rhea endpoint, it returned 18 reactions. Among them

is RHEA:16669, the oxidation of xanthine to urate, whose accession and URI are passed to the UniProt query in the next step. The same pattern of prefixes, a `SELECT` clause, and a graph pattern applies to the other endpoints. The remaining queries are provided in the supplementary material.

Each element in `rows` contains the values returned for the variables in the `SELECT` statement. The required URI or identifier can then be read from one response and used in the next query. Running the steps as one script makes the workflow easier to repeat and inspect. For each step, the endpoint, query, number of returned rows, and identifier passed forward can be recorded. Public endpoints and their contents may change over time, so the results correspond to a particular execution of the workflow. The complete Python code, SPARQL queries, execution log, example output, and exercise material are archived on Figshare [36].

The workflow combines information distributed across several resources. A molecular substructure identifies candidate compounds, one compound is linked to a reaction, the reaction to a protein, the protein to a gene, and the gene to tissue-expression data. Shared identifiers provide the links between these steps.

A query may also return no result. This does not necessarily mean that the query is wrong. An identifier may be incorrect or represented differently, or the requested biological relationship may simply not be present in the resource. The caffeine example shows this directly because its recorded Rhea reactions do not lead to a reviewed human catalyst. When a query returns no rows, the identifier and query conditions should first be checked. If they are correct, the absence itself is part of the result. An automated workflow should report that absence rather than replace it with a guessed identifier or a plausible-looking answer.

10 Conclusion

This tutorial used caffeine as a common thread to show how the same chemical entity can take very different digital forms depending on what needs to be done with it. A compact molecular string is useful for exchange and searching. A structure record can preserve atoms, bonds, coordinates, and properties. Standard identifiers make it easier to recognise the same compound across different resources. Databases organise larger collections, while RDF and knowledge graphs make relationships between compounds, reactions, proteins, genes, and other entities easier to follow. The worked example showed how these links can be used across several public resources to move from a chemical substructure to biological information.

The main lesson is that no representation is suitable for every purpose. Each format preserves some information and leaves other information out. Problems often appear later, when records are compared, converted, or combined, so choices made at the beginning of a workflow can have consequences much further downstream. Careful use of identifiers, consistent data curation, and explicit relationships makes chemical information easier to reuse and easier to check. These foundations also matter for more advanced forms of digital chemistry, including automated reasoning, machine learning, and language-model-

assisted workflows, because such methods are only as reliable as the information they are given. The scope of this tutorial is limited. It follows a single molecule, caffeine, through one workflow, and the worked example reflects the state of the public endpoints at the time it was run. Other molecules, such as polymers, organometallic compounds, or mixtures, and other resources would raise further representation problems that are not covered here.

Author contributions

A.K. conceived and wrote the article and approved the final manuscript.

Conflicts of interest

The author declares no competing interests.

Data availability

No new experimental or computational datasets were generated for this article. The worked examples use publicly available resources cited in the text.

Code availability

The supplementary code, execution log, example output, and exercise material are archived on Figshare (DOI 10.6084/m9.figshare.33473980). The Python code is released under the MIT Licence.

Acknowledgements

No research funding was received for this work. The author thanks the TU Graz Open Access Publishing Fund, and TU Graz and TU Graz Educational Technology for institutional support and training related to the teaching material on which this tutorial is based. Generative AI (ChatGPT) was used for language editing and improvement of clarity; the author reviewed and verified the final content.

References

[1] J. Bai, S. D. Rihm, A. Kondinski, F. Saluz, X. Deng, G. Brownbridge, S. Mosbach, J. Akroyd, and M. Kraft. `twa`: The world avatar python package for dynamic knowledge graphs and its application in reticular chemistry. *Digit. Discov.*, 4(8): 2123–2135, 2025. doi:10.1039/D5DD00069F.

[2] P. Bansal, A. Morgat, K. B. Axelsen, V. Muthukrishnan, E. Coudert, L. Aimo, N. Hyka-Nouspikel, E. Gasteiger, A. Kerhornou, T. B. Neto, M. Pozzato, M.-C. Blatter, A. Ignatchenko, N. Redaschi, and A. Bridge. Rhea, the reaction knowledgebase in 2022. *Nucleic Acids Res.*, 50(D1):D693–D700, 2022. doi:10.1093/nar/gkab1016.

[3] F. B. Bastian, A. Brandulas Cammarata, S. Carsanaro, H. Detering, W.-T. Huang, S. Joye, A. Niknejad, M. Nyamari, T. Mendes de Farias, S. Moretti, M. Tzivanopoulou, J. Wollbrett, and M. Robinson-Rechavi. Bgee in 2024: Focus on curated single-cell RNA-seq datasets, and query tools. *Nucleic Acids Res.*, 53(D1): D878–D885, 2025. doi:10.1093/nar/gkae1118.

[4] A. P. Bento, A. Hersey, E. Félix, G. Landrum, A. Gaulton, F. Atkinson, L. J. Bellis, M. De Veij, and A. R. Leach. An open source chemical structure curation pipeline using RDKit. *J. Cheminform.*, 12:51, 2020. doi:10.1186/s13321-020-00456-1.

[5] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic Acids Res.*, 28(1):235–242, 2000. doi:10.1093/nar/28.1.235. URL https://doi.org/10.1093/nar/28.1.235.

[6] T. Berners-Lee. Linked data. W3C Design Note, July 2006. URL https://www.w3.org/DesignIssues/LinkedData.html. Accessed: 2026-04-27.

[7] C. Bizer, T. Heath, and T. Berners-Lee. Linked data — the story so far. *Int. J. Semant. Web Inf. Syst.*, 5(3):1–22, 2009. ISSN 1552-6283. doi:10.4018/jswis.2009081901.

[8] R. A. Bourne, K. K. M. Hii, and B. J. Reizman. Introduction to synthesis 4.0: towards an internet of chemistry. *React. Chem. Eng.*, 4:1504–1505, 2019. doi:10.1039/C9RE90048A.

[9] S. Bräse. Digital chemistry: navigating the confluence of computation and experimentation – definition, status quo, and future perspective. *Digit. Discov.*, 3:1923–1932, 2024. doi:10.1039/D4DD00130C.

[10] D. Brickley and R. Guha. RDF schema 1.1. W3C Recommendation, World Wide Web Consortium (W3C), 2014. URL https://www.w3.org/TR/rdf-schema/. Accessed 2026-04-30.

[11] I. D. Brown and B. McMahon. CIF: the computer language of crystallography. *Acta Crystallogr., Sect. B*, 58(3):317–324, 2002. doi:10.1107/S0108768102003464. URL https://doi.org/10.1107/S0108768102003464.

[12] CAS. Essential historic chemistry, 2026. URL https://www.cas.org/academic-resource-content/essential-historic-chemistry. Describes ChemZent and Chemisches Zentralblatt coverage from 1830 to 1969.

[13] A. Cereto-Massague, M. J. Ojeda, C. Valls, M. Mulero, S. Garcia-Vallvé, and G. Pujadas. Molecular fingerprint similarity search in virtual screening. *Methods*, 71: 58–63, 2015. doi:10.1016/j.ymeth.2014.08.005.

[14] R. Cyganiak, D. Wood, and M. Lanthaler. RDF 1.1 concepts and abstract syntax. W3C Recommendation, World Wide Web Consortium (W3C), 2014. URL https://www.w3.org/TR/rdf11-concepts/. Accessed: 2026-04-27.

[15] A. Dalby, J. G. Nourse, W. D. Hounshell, A. K. I. Gushurst, D. L. Grier, B. A. Leland, and J. Laufer. Description of several chemical structure file formats used by computer programs developed at molecular design limited. *J. Chem. Inf. Comput. Sci.*, 32(3):244–255, 1992. doi:10.1021/ci00007a012. URL https://doi.org/10.1021/ci00007a012.

[16] K. Degtyarenko, P. de Matos, M. Ennis, J. Hastings, M. Zbinden, A. McNaught, R. Alcántara, M. Darsow, M. Guedj, and M. Ashburner. ChEBI: a database and ontology for chemical entities of biological interest. *Nucleic Acids Res.*, 36(Database issue):D344–D350, 2008. ISSN 0305-1048. doi:10.1093/nar/gkm791.

[17] S. C. Dyer, O. Austine-Orimoloye, A. G. Azov, M. Barba, I. Barnes, V. P. Barrera-Enriquez, A. Becker, R. Bennett, M. Beracochea, A. Berry, J. Bhai, S. K. Bhurji, S. Boddu, P. R. Branco Lins, L. Brooks, S. Budhanuru Ramaraju, L. I. Campbell, M. Carbajo Martinez, M. Charkhchi, L. A. Cortes, C. Davidson, S. Denni, K. Dodiya, S. Donaldson, B. El Houdaigui, T. El Naboulsi, O. Falola, R. Fatima, T. Genez, J. Gonzalez Martinez, T. Gurbich, M. Hardy, Z. Hollis, T. Hunt, M. Kay, V. Kaykala, D. Lemos, D. Lodha, N. Mathlouthi, G. A. Merino, R. Merritt, L. P. Mirabueno, A. Mushtaq, S. N. Hossain, J. G. Pérez-Silva, M. Perry, I. Piližota, D. Poppleton, I. Prosovetskaia, S. Raj, A. I. Abdul Salam, S. Saraf, N. Saraiva-Agostinho, S. Sinha, B. Sipos, V. Sitnik, E. Steed, M.-M. Suner, L. Surapaneni, K. Sutinen, F. F. Tricomi, I. Tsang, D. Urbina-Gómez, A. Veidenberg, T. A. Walsh, N. L. Willhoft, J. Allen, J. Alvarez-Jarreta, M. Chakiachvili, J. Cheema, J. Batista da Rocha, N. H. De Silva, S. Giorgetti, L. Haggerty, G. R. Ilsley, J. Keatley, J. E. Loveland, B. Moore, J. M. Mudge, G. Naamati, J. Tate, S. J. Trevanion, A. Winterbottom, B. Flint, A. Frankish, S. E. Hunt, R. D. Finn, M. A. Freeberg, P. W. Harrison, F. J. Martin, and A. D. Yates. Ensembl 2025. *Nucleic Acids Res.*, 53(D1):D948–D957, 2025. doi:10.1093/nar/gkae1071.

[18] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. Pearson, Boston, MA, 7 edition, 2016. ISBN 978-0-13-397077-7.

[19] G. Fu, C. Batchelor, M. Dumontier, J. Hastings, E. Willighagen, and E. E. Bolton. PubChemRDF: towards the semantic annotation of PubChem compound and substance databases. *J. Cheminform.*, 7(1):34, 2015. ISSN 1758-2946. doi:10.1186/s13321-015-0084-4.

[20] J. Galgonek, J. Vondrášek, and M. Kratochvíl. IDSM ChemWebRDF: SPARQLing small-molecule datasets. *J. Cheminform.*, 13(1):38, 2021. doi:10.1186/s13321-021-00515-1.

[21] D. Geleta, A. Nikolov, G. Edwards, A. Gogleva, R. Jackson, E. Jansson, A. Lamov, S. Nilsson, M. Pettersson, V. Poroshin, B. Rozemberczki, T. Scrivener, M. Ughetto, and E. Papa. Biological insights knowledge graph: An integrated knowledge graph to support drug development. *bioRxiv*, 2021. doi:10.1101/2021.10.28.466262. Preprint.

[22] G. V. Gkoutos, P. Murray-Rust, H. S. Rzepa, and M. Wright. Chemical markup, XML, and the world-wide web. 3. toward a signed semantic chemical web of trust. *J. Chem. Inf. Comput. Sci.*, 41(5):1124–1130, 2001. doi:10.1021/ci000406v. URL https://doi.org/10.1021/ci000406v.

[23] L. Guasch, M. Sitzmann, and M. C. Nicklaus. Enumeration of ring–chain tautomers based on SMIRKS rules. *J. Chem. Inf. Model.*, 54(9):2423–2432, 2014. doi:10.1021/ci500363p. URL https://doi.org/10.1021/ci500363p.

[24] S. R. Hall, F. H. Allen, and I. D. Brown. The crystallographic information file (CIF): A new standard archive file for crystallography. *Acta Crystallogr., Sect. A*, 47 (6):655–685, 1991. doi:10.1107/S010876739101067X. URL https://doi.org/10.1107/S010876739101067X.

[25] K. A. Hamill, R. D. Nelson, G. G. Vander Stouw, and R. E. Stobaugh. Chemical abstracts service chemical registry system. 10. registration of substances from pre-1965 indexes of chemical abstracts. *J. Chem. Inf. Comput. Sci.*, 28(4):175–179, 1988. doi:10.1021/ci00060a002.

[26] S. Harris, A. Seaborne, and E. Prud'hommeaux. SPARQL 1.1 query language. W3C Recommendation, World Wide Web Consortium (W3C), 2013. URL https://www.w3.org/TR/sparql11-query/. Accessed: 2026-04-27.

[27] O. Hartig, P.-A. Champin, G. Kellogg, and A. Seaborne. RDF-star and SPARQL-star. W3C Community Group Final Report, 2021. URL https://www.w3.org/2021/12/rdf-star.html. Accessed 2026-04-30.

[28] J. Hastings, L. Chepelev, E. Willighagen, N. Adams, C. Steinbeck, and M. Dumontier. The chemical information ontology: Provenance and disambiguation for chemical data on the biological semantic web. *PLoS One*, 6(10):e25513, 2011. doi:10.1371/journal.pone.0025513.

[29] J. Hastings, G. Owen, A. Dekker, M. Ennis, N. Kale, V. Muthukrishnan, S. Turner, N. Swainston, P. Mendes, and C. Steinbeck. ChEBI in 2016: improved services and an expanding collection of metabolites. *Nucleic Acids Res.*, 44(D1):D1214–D1219, 2016. ISSN 0305-1048. doi:10.1093/nar/gkv1031.

[30] S. Heller, A. McNaught, S. Stein, D. Tchekhovskoi, and I. Pletnev. InChI—the worldwide chemical structure identifier standard. *J. Cheminform.*, 5:7, 2013. doi:10.1186/1758-2946-5-7. URL https://doi.org/10.1186/1758-2946-5-7.

[31] S. R. Heller, A. McNaught, I. Pletnev, S. Stein, and D. Tchekhovskoi. InChI, the IUPAC international chemical identifier. *J. Cheminform.*, 7:23, 2015. doi:10.1186/s13321-015-0068-4. URL https://doi.org/10.1186/s13321-015-0068-4.

[32] S. Herres-Pawlis, O. Koepler, and C. Steinbeck. Nfdi4chem: Shaping a digital and cultural change in chemistry. *Angew. Chem. Int. Ed*, 58(32):10766–10768, 2019. doi:10.1002/anie.201907260.

[33] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. OWL 2 web ontology language primer. W3C Recommendation, World Wide Web Consortium (W3C), 2012. URL https://www.w3.org/TR/owl2-primer/. Accessed: 2026-04-30.

[34] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4):71:1–71:37, 2021. ISSN 0360-0300. doi:10.1145/3447772.

[35] IUPAC. IUPAC SMILES+. IUPAC Project 2019-002-2-024. URL https://github.com/IUPAC/IUPAC_SMILES_plus. Accessed 2026-04-30.

[36] A. Kondinski. Chemical federated query tutorial, 2026. URL https://doi.org/10.6084/m9.figshare.33473980.

[37] A. Kondinski, J. Bai, S. Mosbach, J. Akroyd, and M. Kraft. Knowledge engineering in chemistry: From expert systems to agents of creation. *Acc. Chem. Res.*, 56(2): 128–139, 2023. doi:10.1021/acs.accounts.2c00617.

[38] M. Kratochvíl, J. Vondrášek, and J. Galgonek. Interoperable chemical structure search service. *J. Cheminform.*, 11(1):45, 2019. doi:10.1186/s13321-019-0367-2.

[39] M. Krenn, F. Hase, A. Nigam, P. Friederich, and A. Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Mach Learn Sci Technol*, 1(4):045024, 2020. doi:10.1088/2632-2153/aba947. URL https://doi.org/10.1088/2632-2153/aba947.

[40] M. Krenn, Q. Ai, S. Barthel, N. Carson, A. Frei, N. C. Frey, P. Friederich, T. Gaudin, A. A. Gayle, K. M. Jablonka, R. F. Lameiro, D. Lemm, A. Lo, S. M. Moosavi, J. M. Nápoles-Duarte, A. Nigam, R. Pollice, K. Rajan, U. Schatzschneider, P. Schwaller, M. Skreta, B. Smit, F. Strieth-Kalthoff, C. Sun, G. Tom, G. F. von Rudorff, A. Wang, A. White, A. Young, R. Yu, and A. Aspuru-Guzik. SELFIES and the future of molecular string representations. *Patterns*, 3(10):100588, 2022. doi:10.1016/j.patter.2022.100588.

[41] J.-B. Lamy. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artif. Intell. Med.*, 80:11–28, 2017. doi:10.1016/j.artmed.2017.07.002.

[42] G. Landrum et al. Rdkit: Open-source cheminformatics. URL https://www.rdkit.org/. Accessed 2026-04-08.

[43] D. J. Lipman and W. R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985. doi:10.1126/science.2983426. URL https://doi.org/10.1126/science.2983426.

[44] T. Lombardot, A. Morgat, K. B. Axelsen, L. Aimo, N. Hyka-Nouspikel, A. Niknejad, A. Ignatchenko, I. Xenarios, E. Coudert, N. Redaschi, and A. Bridge. Updates in Rhea: SPARQLing biochemical reaction data. *Nucleic Acids Res.*, 47(D1):D596–D600, 2019. doi:10.1093/nar/gky876.

[45] J. McLaughlin, K. Lagrimas, Z. Pendlington, C. Casals-Casas, G. Owen, S. Jupp, and H. Parkinson. OLS4: A new ontology lookup service for a growing interdisciplinary knowledge ecosystem. *Bioinformatics*, 41(5):btaf279, 2025. doi:10.1093/bioinformatics/btaf279.

[46] D. Mendez, A. Gaulton, A. P. Bento, J. Chambers, M. De Veij, E. Félix, M. P. Magariños, J. F. Mosquera, P. Mutowo, M. Nowotka, M. Gordillo-Marañón, F. Hunter, L. Junco, G. Mugumbate, M. Rodriguez-Lopez, F. Atkinson, N. Bosc, C. J. Radoux, A. Segura-Cabrera, A. Hersey, and A. R. Leach. ChEMBL: towards direct deposition of bioassay data. *Nucleic Acids Res.*, 47(D1):D930–D940, 2019. ISSN 0305-1048. doi:10.1093/nar/gky1075.

[47] A. Menon, N. B. Krdzavac, and M. Kraft. From database to knowledge graph: Using data in chemistry. *Curr. Opin. Chem. Eng.*, 26:33–37, 2019. doi:10.1016/j.coche.2019.08.004.

[48] P. Murray-Rust and H. S. Rzepa. Chemical markup, XML, and the world wide web. 1. basic principles. *J. Chem. Inf. Comput. Sci.*, 39(6):928–942, 1999. doi:10.1021/ci990052b. URL https://doi.org/10.1021/ci990052b.

[49] P. Murray-Rust and H. S. Rzepa. Chemical markup, XML and the world-wide web. 2. information objects and the CMLDOM. *J. Chem. Inf. Comput. Sci.*, 41(5):1113–1123, 2001. doi:10.1021/ci000404a. URL https://doi.org/10.1021/ci000404a.

[50] M. A. Musen and Protégé Team. The Protégé project: A look back and a look forward. *AI Matters*, 1(4):4–12, 2015. doi:10.1145/2757001.2757003.

[51] N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison. Open babel: An open chemical toolbox. *J. Cheminform.*, 3:33, 2011. doi:10.1186/1758-2946-3-33.

[52] OpenSMILES. OpenSMILES specification. URL http://opensmiles.org/opensmiles.html. Blue Obelisk community project. Accessed 2026-04-30.

[53] W. R. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. U.S.A.*, 85(8):2444–2448, 1988. doi:10.1073/pnas.85.8.2444.

[54] I. Pletnev, A. Erin, A. McNaught, K. Blinov, D. Tchekhovskoi, and S. Heller. InChIKey collision resistance: an experimental testing. *J. Cheminform.*, 4:39, 2012. doi:10.1186/1758-2946-4-39. URL https://doi.org/10.1186/1758-2946-4-39.

[55] E. Prud'hommeaux and G. Carothers. RDF 1.1 turtle: Terse RDF triple language. W3C Recommendation, World Wide Web Consortium (W3C), 2014. URL https://www.w3.org/TR/turtle/. Accessed: 2026-04-30.

[56] D. Rogers and M. Hahn. Extended-connectivity fingerprints. *J. Chem. Inf. Model.*, 50(5):742–754, 2010. doi:10.1021/ci100050t.

[57] C. Steinbeck, Y. Han, S. Kuhn, O. Horlacher, E. Luttmann, and E. Willighagen. The chemistry development kit (CDK): An open-source Java library for chemo- and bioinformatics. *J. Chem. Inf. Comput. Sci.*, 43(2):493–500, 2003. doi:10.1021/ci025584y.

[58] The UniProt Consortium. UniProt: The universal protein knowledgebase in 2025. *Nucleic Acids Res.*, 53(D1):D609–D617, 2025. doi:10.1093/nar/gkae1010.

[59] D. Vrandečić and M. Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014. doi:10.1145/2629489.

[60] D. Weininger. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36, 1988. doi:10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

[61] D. Weininger, A. Weininger, and J. L. Weininger. SMILES. 2. algorithm for generation of unique SMILES notation. *J. Chem. Inf. Comput. Sci.*, 29(2):97–101, 1989. doi:10.1021/ci00062a008. URL https://doi.org/10.1021/ci00062a008.

[62] D. W. Weisgerber. Chemical abstracts service chemical registry system: History, scope, and impacts. *J. Am. Soc. Inf. Sci.*, 48(4):349–360, 1997. doi:10.1002/(SICI)1097-4571(199704)48:4<349::AID-ASI8>3.0.CO;2-W.

[63] D. S. Wigh, J. M. Goodman, and A. A. Lapkin. A review of molecular representation in the age of machine learning. *WIREs Comput. Mol. Sci.*, 12(5):e1603, 2022. doi:10.1002/wcms.1603. URL https://doi.org/10.1002/wcms.1603.

[64] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons. The fair guiding principles for scientific data management and stewardship. *Sci. Data*, 3:160018, 2016. doi:10.1038/sdata.2016.18.