atoms

MDPI

Article

CollisionDB: A New Database of Atomic and Molecular Collisional Processes with an Interactive API

Christian Hill [1,*], Dipti [1], Kalle Heinola [1] and Martin Haničinec [2,3]

[1] International Atomic Energy Agency, 1400 Vienna, Austria
[2] Department of Physics and Astronomy, University College London, London WC1E 6BT, UK
[3] digid GmbH, 55129 Mainz, Germany
* Correspondence: fusion-data@iaea.org

Abstract: The Atomic and Molecular Data Unit of the International Atomic Energy Agency has developed a new database, CollisionDB, to provide an open, free, robust and long-term repository of data on plasma collisional processes. The database contains data on cross sections and rate coefficients for collisions of electrons, photons and heavy particles with atomic and molecular species. A fundamental requirement for this database is the implementation of standardized metadata, which provide an unambiguous description of the collisional data available in peer-reviewed sources. CollisionDB offers both a browser-based search interface and an application programming interface (API) that allows users to filter, process and compare collisional datasets. For this purpose, a Python package `PyCollisionDB` has been developed to access the CollisionDB API. Here, we present an overview of the technical developments, including data schemas, standards and user interface underlying the CollisionDB application, with particular emphasis on the API developed to support the integration of data into modeling and other codes.

Keywords: plasma physics; atomic and molecular data; databases; collisional cross sections; collisional rate coefficients

Citation: Hill, C.; Dipti; Heinola, K.; Haničinec, M. CollisionDB: A New Database of Atomic and Molecular Collisional Processes with an Interactive API. *Atoms* **2024**, *12*, 20. https://doi.org/10.3390/atoms12040020

Academic Editor: Kanti M. Aggarwal

Received: 29 February 2024
Revised: 20 March 2024
Accepted: 21 March 2024
Published: 27 March 2024

1. Introduction

Plasma processes are central to many areas of scientific and technical research and development. Modeling such processes requires accurate cross section and/or rate coefficient datasets for a wide variety of collisional processes. To this end, many experimental and theoretical studies of such processes are described in the peer-reviewed scientific literature each year. Although individual journals may make these data available in some form, the datasets are usually more conveniently obtained through one of several actively developed databases, most of which focus on a particular application, including LXCat (cold plasma) [1], the database of the Data Center for Plasma Properties at the Korea Institute of Fusion Energy (fusion energy research) [2], Phys4Entry (to model re-entry plasma in planetary atmospheres) [3], QuantemolDB (focused on plasma chemistry for technological applications) [4] and ADAS (rate coefficients and codes for astrophysics and fusion energy applications) [5]. These databases provide different online interfaces for searching and retrieving data and, generally, adopt different conventions for describing the species, states and processes involved.

The Atomic and Molecular Data (AMD) Unit of the International Atomic Energy Agency (IAEA) has, since 1988, maintained the ALADDIN database of evaluated data for fusion energy research [6]. It too has an online search interface (https://www-amdis.iaea.org/ALADDIN/, accessed on 23 March 2024) and provides the coefficients and details of functional fits to key processes. However, its datasets are not all described by unambiguous metadata, and it does not expose an Application Programming Interface (API) for the automated searching and retrieval of data.

CollisionDB is an open-source repository for cross section and rate coefficient data of plasma collisional processes, developed to support fusion research with an emphasis on the FAIR (Findable, Accessible, Interoperable and Reusable) principles [7] of database management. All data hosted in CollisionDB are associated with rich metadata needed to describe collisional datasets and are assigned a persistent, globally unique identifier (F). Data can be retrieved by searching both from the user interface and through the API (A). In order to facilitate the interpretation and exchange of datasets (I), data are provided in a formal, accessible and standardized format. Furthermore, including provenance information, such as a DOI, enhances the comprehensibility and traceability of the data. Data are made available under a clear license with relevant attributes and unambiguous metadata to ensure reusability (R).

While the data that CollisionDB contains are of most relevance to fusion energy research, the standards, formats and software stack it is built around, which are described in this article, should prove of use for similar databases of collisional and chemical reaction processes for other applications. In particular, the data model and API described below allow the automated querying and retrieval of data and metadata directly from code in a way that facilitates visualization, data exploration and machine learning.

2. Results

2.1. Search Interface

CollisionDB offers a user-friendly web interface to interact with the database, implemented with various features such as filtering, retrieval and easy access to datasets. Figure 1 shows the search form for querying the database based on a set of relevant metadata attributes describing the collisional datasets. Users can search for reactants or products as species (with or without states), and Table 1 provides examples of species and state notation used in CollisionDB. Other search options include method, data type and process types, which can be selected from the appropriate drop-down menu. The search can be further refined by filtering for author and publication DOI. Additionally, the users can retrieve the evaluated/recommended data from the search form. Since the datasets are timestamped, users can also retrieve deprecated datasets using the `valid_on` date field from the search form. In this way, a query on a known date can always be reproduced exactly, with the same datasets returned as were originally downloaded.

Table 1. Examples of species and states identification using PyValem notation in CollisionDB.

Species or State	PyValem Notation	Chemical Notation
Atoms	`Li`, `Be`, `W`	
Molecules	`H2`, `LiH`, `H2O`	H_2, LiH, H_2O
Ions	`H+`	H^+
	`Be+4`	Be^{4+}
	`H2-`	H_2^-
	`CO3-2`	CO_3^{2-}
Isotopes	`(2H)` or `D`	^2H (D)
	`(6Li)`	^6Li
	`(235U)`	^{235}U
Isotopologues	`(2H)2` or `D2`	$^2H_2$ (D_2)
	`(13C)H4`	$^{13}CH_4$
Atomic configurations	`Li 1s2.2s` or `Li [He].2s`	Li $1s^22s$
	`Ne+ 1s2.2s2.2p5`	Ne^+ $1s^22s^22p^5$
Atomic term symbols	`He 1S`	He ^1S
	`Al 2P_3/2`	Al $^2P_{3/2}$
Molecular configurations	`H2 1σg2` or `H2 1sigmag2`	H_2 $1\sigma_g^2$
	`H2+ 1sigmag`	H_2^+ $1\sigma_g$
	`Be2+ 1sigma2.2pi`	Be_2^+ $1\sigma^2 2\pi$
Molecular term symbols	`O2 X(3SIGMA-g)`	O_2 $X(^3\Sigma_g^-)$
	`AlO C(2PI)`	AlO $C(^2\Pi)$
Quantum numbers, labels	`n=4`, `|m|=1`, `par=+`	

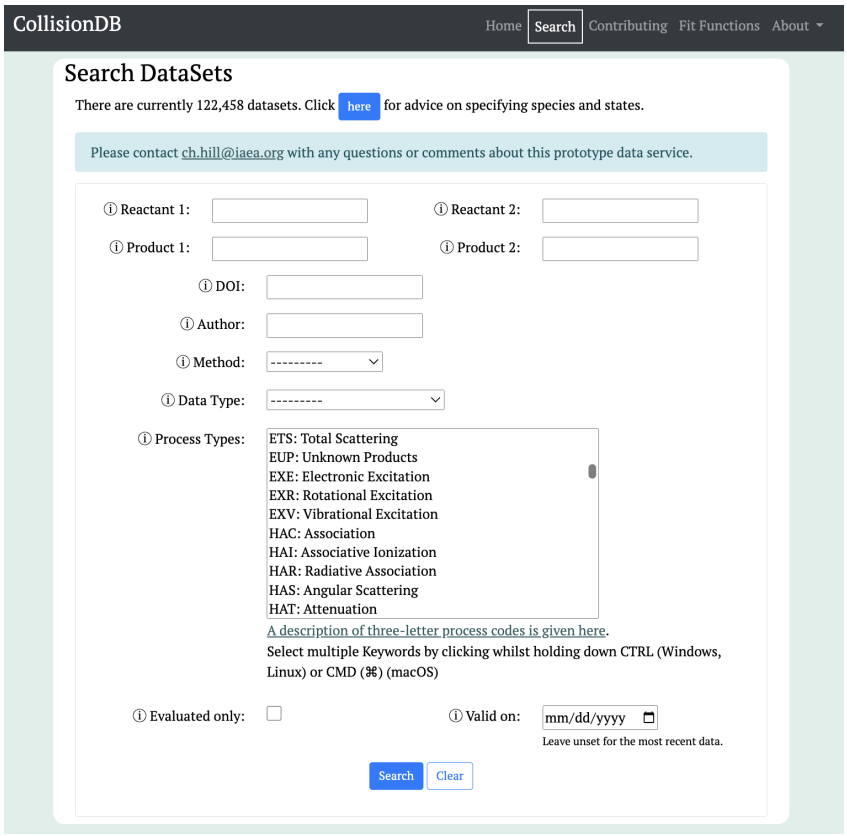

Figure 1. The browser-based search form for querying data in CollisionDB, available online at https://amdis.iaea.org/db/collisiondb/search/ (accessed on 23 March 2024).

The datasets that match the search query will be displayed in a paginated list on the search results page. Users can download individual datasets as plain-text files (.txt) (including the JSON metadata header) as shown in Listing 1. Alternatively, users can download all datasets as an archive, which includes individual dataset files, a manifest file and a bibliography file: see the archive structure in Figure 2. The manifest file provides the list of dataset files within the archive, identified by their "qualified ID" (`qid`; see Table 2). It also includes additional information about the download timestamp, a universally unique identifier for the archive name (`uuid`), the query string (`GET_string`) and number of datasets. A sample manifest file is shown in Listing 2, demonstrating a search for `reactant1` (Be+4) as extracted from `GET_string`, along with information about the retrieved archive.

Listing 1. An example dataset file for download.

```
{
    "qid": "D15115",
    "reaction": "e- + HD X(1Σ+g);v=0 → HD C(3Πu);v=0 + e-",
    "process_types": {
        "EXE": "Electronic Excitation",
        "EXV": "Vibrational Excitation"
    },
    "data_type": "cross section",
    "refs": {
        "B9": {
            "doi": "10.1016/j.adt.2020.101403"
        }
    },
    "comment": "MCCC calculations of vibrationally-resolved electronic excitation of HD,
         adiabatic nuclei calculations performed with the spheroidal MCCC(210) model",
    "method": "MCCC",
    "columns": [
            {
```

```
                "name": "E",
                "units": "eV"
            },
            {
                "name": "sigma",
                "units": "cm2"
            }
    ],
    "unc_perc": 10.0,
    "threshold": 12.3069,
    "fit": {
        "coeffs": {
            "A1": 0.6196,
            "A2": 0.70754,
            "A3": -4.2742,
            "A4": 9.8038,
            "A5": -9.5964,
            "A6": 3.4918
        },
        "func": "singlet_singlet_H2"
    },
    "data_from_fit": false,
    "metadata_version": "M1.0",
    "time_added": "2022-05-26 10:37:10.730416+00:00"
}
------------------------------------------------------------------------
1.231e+01 0.0
1.250e+01 1.775e-20
1.300e+01 1.568e-19
1.350e+01 2.945e-19
```


Listing 2. A sample manifest JSON file describing the dataset files in the archive.


```
{
    "timestamp": "2023-06-23 10:03:01.071468+00:00",
    "uuid": "129bbb33-ad86-4f92-9128-4dffa6c53476",
    "GET_string": "reactant1=Be%2B4",
    "ndatasets": 68,
    "datasets": {
        "D76333": {
                    "reaction": "Be+4 + H 1s → Be+3 + H+",
                    "refs": ["B20"]
                    }
        "D76335": {
                    "reaction": "Be+4 + H 1s → Be+4 + H+ + e-",
                    "refs": ["B20"]
                    }
         .....
         .....
    }
}
```


```
Data Directory
└── 129bbb33-ad86-4f92-9128-4dffa6c53476.zip
    ├── 76333.txt
    ├── 76335.txt
    ├── ....
    ├── manifest.json
    └── bibliography.json
```

Figure 2. Archive structure for downloaded data.

Table 2. A description of the JSON metadata header used in each dataset file.

Key	Description
`qid`	For downloaded dataset files, the qualified ID of the dataset: the primary key integer prefixed with `D`.
`reaction`	A collisional process between stateful species described by the conventions of the PyValem Python package [8].
`process_types`	A sequence of three-letter codes describing the collisional process [9] and a brief description.
`data_type`	Type of the collisional data, one of `"cross section"`, `"rate coefficient"` or `"differential cross section"`.
`refs`	A sequence of key–value pairs identifying the references cited for this dataset; the key is a qualified ID for an entry in the CollisionDB `refs_ref` table (an integer prefixed with the letter `B`), and the value is typically a JSON representation of the reference's DOI or URL
`comment`	A free-text field with additional information describing the dataset.
`method`	A general method, which can be one of `"experiment"`, `"semi-empirical"`, `"theory"` or `"estimate"`. Alternatively, details of the computational method used can be provided using the pre-defined abbreviations given online at https://amdis.iaea.org/db/collisiondb/theoretical-methods/ (accessed on 23 March 2024), e.g. `"MCCC"` for molecular convergent close-coupling.
`columns`	A list of JSON objects identifying the names and units of the columns in the numerical data part of the dataset file; the column metadata are ordered in the same way as the columns themselves, and the units are parseable by the PyQn Python library [10].
`unc_perc`	The uncertainty as a relative percentage value of the data, specified as a numerical value. Uncertainties per data point can be provided along with numerical data, as shown in the example Listing 10.
`threshold`	The threshold energy, in eV, for a given transition should be a number (without units).
`frame`	Energy frame of reference, one of `"target"` or `"com"` for target and center-of-mass frames, respectively. Mandatory only for heavy-particle collisions.
`channel`	A comma-separated list of reaction(s) describing the composite process with multiple channels.
`recommended`	Indicates whether the dataset is evaluated and recommended.
`deprecated`	Represents whether the dataset is deprecated.
`data_from_fit`	Mandatory only for datasets with fit coefficients, describes the source of the numerical data, either `false` (data from the original source) or `true` (data derived from fit coefficients).
`fit`	The values of named fitting coefficients as key–value pairs, as well as other data such as the fitting uncertainty (`fit_unc_perc`), the fit function name, `func` and the limits of validity of the fitting function, `Elo` and `Ehi`.
`metadata_version`	Provides the specific version of the metadata schema associated with the dataset.
`time_added`	Indicates the timestamp for when the dataset was added.
`time_deprecated`	Indicates the timestamp for when the dataset was deprecated (if applicable).

The web interface also allows users to access detailed information about individual collisional datasets, along with an interactive graphical display of the data, as demonstrated in Figure 3 for a dataset with the primary key ID `15115`. This collisional dataset contains the fit coefficients along with cross sections, which can also be viewed through the interface (see Figure 4). As can be seen from this figure, the rendered LaTeX representation of the corresponding fit function is shown along with its Python implementation. More

information about these fitting functions is available online at https://amdis.iaea.org/db/collisiondb/fit-functions/ (accessed on 23 March 2024).

DataSet D15115

$$e^- + H^2H\ X^1\Sigma_g^+\ v = 0 \rightarrow H^2H\ C^1\Pi_u\ v = 0 + e^-$$

Process	EXE: Electronic Excitation, EXV: Vibrational Excitation
Data type	cross section | uploaded on 2022-05-26
Comment	MCCC calculations of vibrationally-resolved electronic excitation of HD, adiabatic nuclei calculations performed with the spheroidal MCCC(210) model
Method	MCCC: Molecular convergent close-coupling
Columns	1. E /eV 2. sigma /cm^2
Threshold	12.3069 eV
Uncertainty	10 %
Ref	B9: L. H. Scarlett, D. V. Fursa, M. C. Zammit, I. Bray, Y. Ralchenko, "Complete collision data set for electrons scattering on molecular hydrogen and its isotopologues: II. Fully vibrationally-resolved electronic excitation of the isotopologues of H_2 ($X^1\Sigma_g^+$) ", *Atomic Data and Nuclear Data Tables* **139**, 101403 (2021). [10.1016/j.adt.2020.101403]
Data	Download

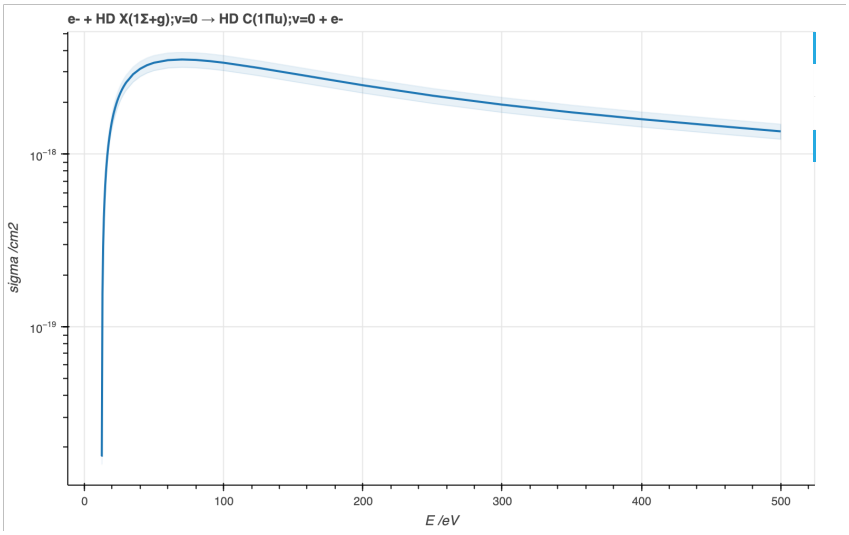

Figure 3. An example showing the detailed information of a collisional dataset, identified by its primary key ID 15115, along with an interactive graphical representation of the collisional data. Available online at https://amdis.iaea.org/db/collisiondb/datasets/15115 (accessed on 23 March 2024).

Fitted Data

Fit Function
Details

$$\sigma(x) = \left| \frac{x-1}{x} \cdot \left(\frac{A_1^2}{x} \ln(x) + \frac{A_2}{x} + \frac{A_3}{x^2} + \frac{A_4}{x^3} + \frac{A_5}{x^4} + \frac{A_6}{x^5} \right) \right|$$

Python

```
def singlet_singlet_H2(x, A1, A2, A3, A4, A5, A6):
    """
    This function calculates the vibrational and dissociative excitation
     cross sections  (in a.u.) of H2 isotopologues from the ground electronic
     state X to singlet excited states.

    param x: requested electron-impact energy in threshold units
    type x: float, np.ndarray
    param Ai: fit coefficients
    type Ai: float
    """
    sigma = ((x-1)/x)*(A1**2 * np.log(x)/x + A2/x + A3/x**2 + A4/x**3 +
                       A5/x**4 + A6/x**5)
    return np.absolute(sigma)
```

Fit Coefficients

A1	6.196e−01
A2	7.075e−01
A3	−4.274e+00
A4	9.804e+00
A5	−9.596e+00
A6	3.492e+00

x-range –

Figure 4. Fit coefficients for collisional dataset (ID=15115) along with the rendered LaTeX representation of the corresponding fit function and its Python implementation. Details of the fit function can be found online at https://amdis.iaea.org/db/collisiondb/fit-functions/singlet_singlet_H2 (accessed on 23 March 2024), identified by the fit function name.

2.2. PyCollisionDB Package for API

`PyCollisionDB` [11] is a Python package for interacting with the CollisionDB API to obtain collisional datasets from the database. Datasets can be retrieved in a standardized way, output in different formats, compared and manipulated using a number of pre-defined Python methods.

To use `PyCollisionDB`, users need to install and import the `PyCollision` module to access the available functions and attributes, as shown in Listing 3.

Listing 3. Initialize the main instance of PyCollisionDB package to interact with the database in Python Shell.

```
>>> # import PyCollision module to access associated functions and attributes.
>>> from pycollisiondb import PyCollision
```

The main methods of the `PyCollisionDB` package, including query structures/schema and usage examples to efficiently explore interaction and data exchange, are described below:

- `PyCollision.get_datasets()`: This is the main class method for querying and retrieving the datasets in a standardized format from the server for a given query, which should be passed as a Python dictionary (`dict` object). Only valid metadata keys, such as `ids` (a list of dataset IDs) or `reactants`, are accepted in each query. The values are specified as either a string or a list of comma-separated strings depending upon the query key. A list of valid keys and examples are given in Table 3.

Table 3. Examples of valid query keys for interacting with CollisionDB database over API.

Key	Examples
pks or ids	[15111]
	[105645, 101678, 789]
reaction_texts	['W+61 + H 1s -> W+60 + H+', 'W+62 + H 1s -> W+61 + H+']
	['e- + LiH X(1SIGMA+g);v=0 -> LiH A(1SIGMA+g);v=10 + e-']
reactant1	'e' or 'e-'
reactant2	'H2'
reactants	['H', '(2H)2']
product1	'Li+'
product2	'Be+2 1s.3s S=0'
products	['W+60 n=25', 'H+']
method	'semi-empirical'
	'experiment'
process_types	['HCX']
	['EXE', 'EXV']
data_type	'cross section'
	'rate coefficient'
doi	'10.1088/1361-6455/ac22e1'
evaluated	True
valid_on	'2022-08-22'

An example of querying the database for the proton-impact ionization cross sections of hydrogen in its ground state using the PyCollisionDB package in a Python shell is given in Listing 4. In this example, the database is queried in two ways: (a) by passing a list of reactants ['H+', 'H 1s'] combined with process types ['HIN'], which refers to heavy-particle impact ionization, and (b) using the query key reaction_texts. Both queries returned the same number of datasets and a zip-compressed archive comprising individual dataset files, a manifest and bibliography file, as explained in the "Search interface" section above.

Listing 4. Search and fetch datasets from the server over the API.

```
>>> # (a) database query using a list of reactants and process_types
>>> query = {
              'reactants': ['H+', 'H 1s'],
              'process_types': ['HIN'],
              'data_type': 'cross section'
            }
>>> pycoll = PyCollision.get_datasets(query=query)
>>> # number of datasets matching the search query
>>> len( pycoll.datasets)
4

>>> # (b) database query using reaction_texts
>>> query = {
              'reaction_texts': ['H+ + H 1s -> H+ + H+ + e-'],
              'data_type': 'cross section'
            }
>>> pycoll = PyCollision.get_datasets(query=query)
>>> len( pycoll.datasets)
4
```

- PyCollision.summarize_datasets(): This method provides a summary of the retrieved datasets and groups them into different blocks based on the reaction text. The output includes pertinent information such as qualified ID, process types, data type and references for each collisional dataset, as shown in Listing 5.

Listing 5. Summary of datasets in blocks for each distinct reaction text.

```
>>> pycoll.summarize_datasets()
H+ + H 1s → H+ + H+ + e-
================================================================================
 qid: D102737
 process_types: ['HIN']
 data_type: cross section
 refs: {'B32': {'doi': '10.1016/j.adt.2019.05.002'}}
 qid: D107356
 process_types: ['HIN']
 data_type: cross section
 refs: {'B45': {'doi': '10.1140/epjd/e2019-100380-x'}}

H 1s + H+ → H+ + H+ + e-
================================================================================
 qid: D103103
 process_types: ['HIN']
 data_type: cross section
 refs: {'B33': {'doi': '10.1088/0022-3700/14/14/009'}}
 qid: D103104
 process_types: ['HIN']
 data_type: cross section
 refs: {'B34': {'doi': '10.1088/0022-3700/20/11/016'}}
```

- `PyCollision.resolve_refs()`: Call this method to resolve the references for all retrieved datasets into a proper, citeable format. The method returns references (`refs`) as a Python dictionary, with bibliographic data identified by reference ID; see Listing 6.

Listing 6. Resolving the references for all the datasets into a proper, citeable format.

```
>>> pycoll.resolve_refs()
>>> pycoll.refs
{'B32': {
         'authors': 'H. Agueny, J. Petter Hansen, A. Dubois, A. Makhoute, A. Taoutioui, N.
             Sisourat',
         'title': 'Electron capture, ionization and excitation cross sections for keV
             collisions between fully stripped ions and atomic hydrogen in ground and
             excited states',
         'journal': 'Atomic Data and Nuclear Data Tables',
         'volume': '129-130',
         'page_start': '101281',
         'page_end': '',
         'article_number': '101281',
         'year': 2019,
         'note': '',
         'doi': '10.1016/j.adt.2019.05.002',
         'bibcode': '2019ADNDT.12901281A',
         'url': 'https://dx.doi.org/10.1016/j.adt.2019.05.002'
            },
 'B33': {...},
 'B34': {...},
 'B45': {...}
}
```

- `PyCollision.datasets`: To access the details of a collisional dataset, users can use the dataset ID as a key for the `datasets` `dict` attribute. An example of retrieving the metadata and numerical data of a specific dataset `102737` is given in Listing 7.

Listing 7. Retrieve details of an individual dataset.

```
>>> # access the metadata by its primary ID
>>> pycoll.datasets[102737].metadata
{'qid': 'D102737',
 'reaction': 'H+ + H 1s → H+ + H+ + e-',
 'process_types': {'HIN': 'Ionization'},
 'data_type': 'cross section',
 'refs': {'B32': {'doi': '10.1016/j.adt.2019.05.002'}},
 'comment': 'Ionization cross sections in H+ + H collisions using a semiclassical close-
     coupling approach. Cross     sections  represent the average values of the results
     obtained with two basis sets and the uncertainties provide the estimate of
     convergence of the cross sections',
 'method': 'CC',
 'columns': [{'name': 'E', 'units': 'eV.u-1'},
             {'name': 'sigma', 'units': 'cm2'}]
}

>>> # Prints the numerical values of dataset along with units.
>>> pycoll.datasets[102737].print_values()
E / eV.u-1 sigma / cm2
1000.0 5.471e-19
4000.0 1.96e-18
9000.0 1.235e-17
..
100000.0 1.248e-16
```

- `PyCollision.convert_units()`: Use this method to change the units for any or all datasets. This function accesses the PyQn library [10] to perform unit conversions. Listing 8 returns all datasets with energy and cross sections in units of keV u^{-1} and Mb, respectively.

Listing 8. Unit conversions of all datasets.

```
>>> # Energy is changed from eV.u-1 to keV.u-1 and sigma from cm2 to Mb.
>>> pycoll.convert_units({'E': 'keV.u-1', 'sigma': 'Mb'})
>>> pycoll.datasets[102737].print_values()
E / keV.u-1 sigma / Mb
1.0   0.5471
4.0   1.9600
9.0  12.35
...
```

- `PyCollision.plot_all_datasets()`: The `PyCollision` module also provides visualization function for the retrieved datasets, which assists in data evaluation and quality assessment. The `plot_all_datasets` method can be used to create plots using the `pyplot` submodule of the Matplotlib library. An example representation of the retrieved datasets for the proton-impact ionization of `H 1s` is shown in Figure 5. One can see that the peak cross sections reported in the references identified within CollisionDB as `B33` [12] and `B34` [13] are about 20–30% lower than those in other works [14,15]. As is evident in Listing 9, data visualization allows users to intuitively identify inconsistencies in the data. This serves as a first step in evaluating the data quality and requires further in-depth analysis of the data [16].

Listing 9. An example of data visualization.

```
>>> import matplotlib.pyplot as plt
>>> # Make a plot, indicating how the data should be labelled.
>>> fig, ax = plt.subplots()
>>> # The default legend consists of dataset qualified ID (qid) and reaction labels;
>>> # it can be customized to include refs and process_types or either of these lables.
>>> pycoll.plot_all_datasets(ax, label=('reaction', 'qid', 'refs', 'process_types'))
>>> plt.xlim(0, 180)
>>> plt.legend()
```

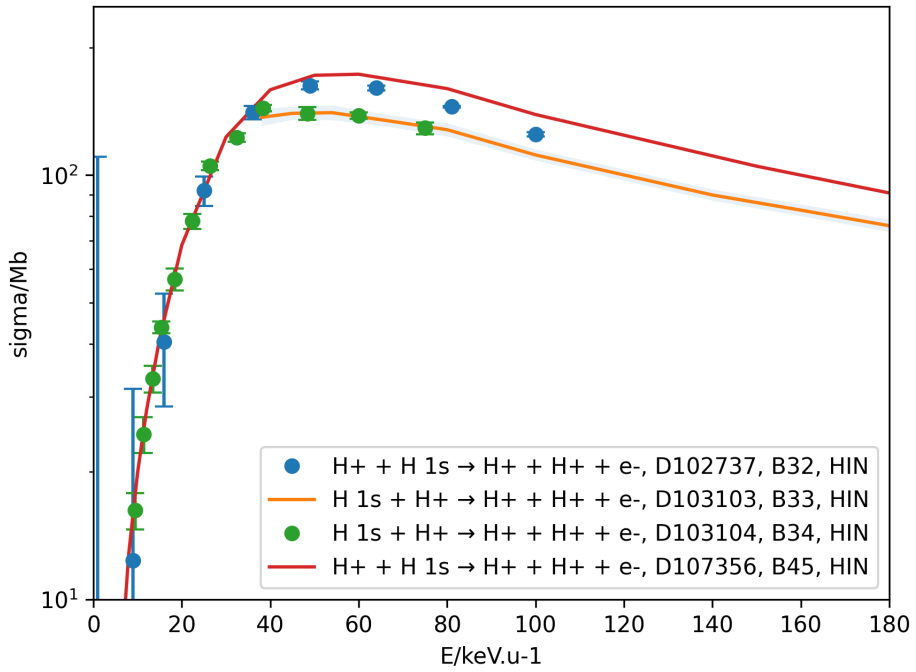

Figure 5. Comparison of the different datasets for proton-impact ionization cross sections of H 1s. Datasets were retrieved from the CollisionDB server using the API, and the legend includes reaction text, dataset qualified ID, reference ID and process code.

In the context of querying the database through the API, filtering is based on the search for the `text` representation of the objects, such as reactants and products, where applicable. However, filtering the datasets by reaction text looks for the `ordered_text` attribute stored in the `rxn_reaction` table, returning all reactions regardless of the order of reactants and products. This is implemented to ensure that there are no missing datasets in the query, as the cross sections typically do not depend on the order of the reactants if the relative velocities are the same. As can be seen in the proton-impact ionization example, the equivalent representations of the reactants `"H+ + H 1s"` or `"H 1s + H+"` represent the same collision process. This is ensured by the fact that energies for heavy-particle collisions are standardized in eV u^{-1}, giving the same relative velocity between the (non-)identical reactants. However, there may be cases where the order of reactants/products can affect the cross sections, in particular, for identical species but with different atomic states. For example, `"H+ + H -> H+ + H 2p"` and `"H+ + H -> H 2p + H+"` represent two different collisional processes, the former being the proton-impact excitation of hydrogen into the excited state and the latter representing the electron capture by protons into the 2p state. These processes can be distinguished by looking at the associated process types with the values `"HEX"` and `"HCX"`, respectively. However, the reactions like `"He 1s.2s 3S + He 1s2 1S -> He 1s.4s 3S + He"` and `"He 1s.2s 3S + He 1s2 1S -> He + He 1s4s 3S"` belong to the same process type `"HEX"`, but they represent two different collisional processes. The first reaction represents the excitation of the projectile He from $1s2s\ ^3S$ to $1s4s\ ^3S$ with a threshold energy of about 4 eV, while the latter is the target excitation from ground state $1s^2\ ^1S$ to $1s4s\ ^3S$ with a threshold energy of about 24 eV. We have preserved the canonical form of the reaction as well as the ordered text to search for all possible equivalent reactions. Additional metadata, such as process types, threshold and comments, can guide the users when affected by reactant/product order.

2.3. Current Status of CollisionDB

As of July 2023 there are 122,352 datasets in CollisionDB. Figure 6 summarizes the distribution of datasets for individual atoms and ions; as can be seen from this figure, there

are some data for almost all charge states of atoms lighter than iron, but for heavier species there are data only for either neutral or low-charge states or for nearly fully stripped ions. There is better coverage for xenon and tungsten because of their importance for magnetic confinement fusion experiments.

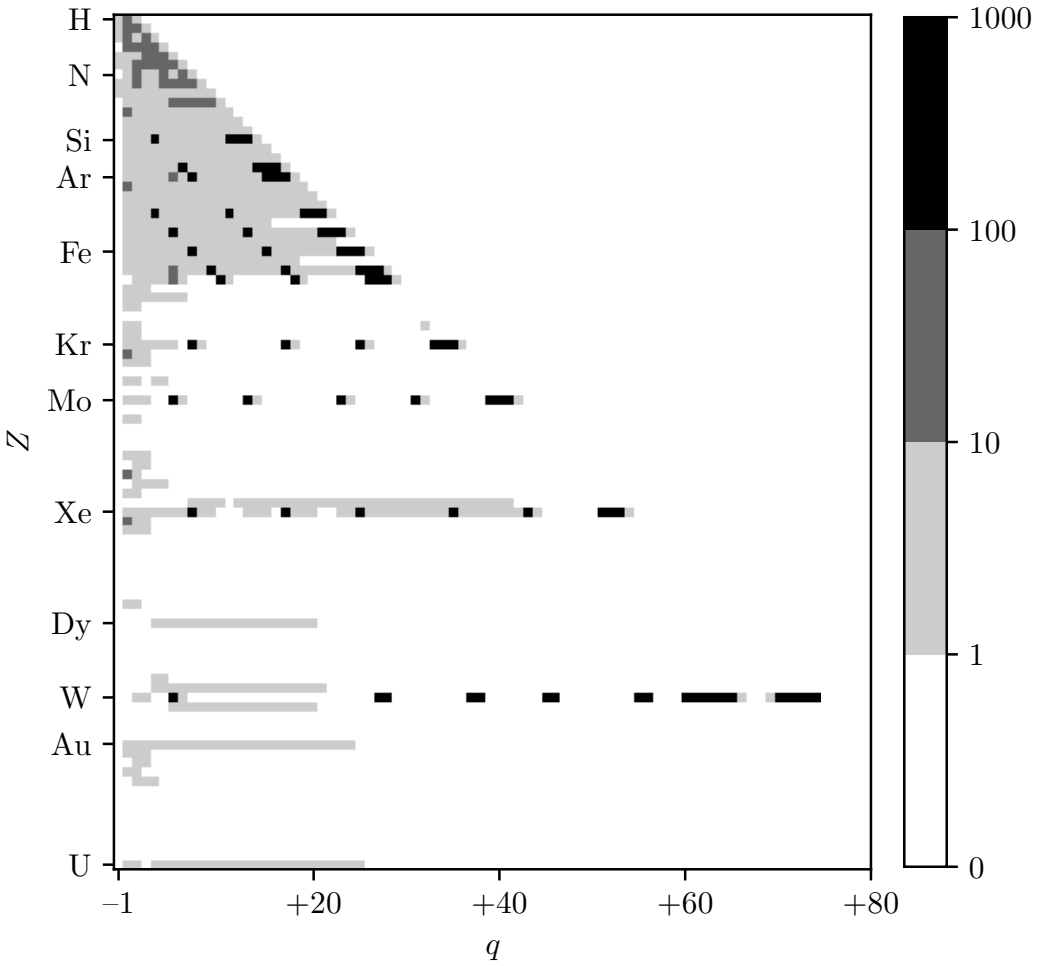

Figure 6. A summary of the number of collisional datasets available in CollisionDB as of July 2023 for atomic species: Z is the nuclear charge and q the ion charge.

A breakdown of the classification of datasets in CollisionDB by process type and reactant type (total vs. molecular species) is given in Table 4. The database is dominated by the large number of electron-impact vibronic excitation cross sections and rate coefficients for molecular hydrogen and its isotopologues. At the time of writing, there are 115,773 cross sections and 6579 rate coefficient datasets in CollisionDB. Nearly 60% of the datasets have been fitted to functions described within CollisionDB, each of them with a Python implementation and some also in Fortran. These fit functions can be used on provided energy or temperature intervals to generate and return custom datasets.

A release of all datasets as a downloadable archive will be made on a periodic basis; the first such release, 2023.1, was issued on 31 August 2023 and is available from the CollisionDB website (see Data availability, below).

Table 4. Reactions categorized by collisional process type and by reactant as molecule. Abbreviations starting with the letter `E` denote "electron-impact" processes; those starting with the letter `H` are "heavy-particle" collisions.

Abbreviation	Process Type	Total Number of Reactions	Number Involving Molecular Reactants
`COM`	Composite Process with Multiple Channels	1107	1087
`EDA`	Dissociative Attachment	93	93
`EDE`	Dissociative Excitation	2380	2380
`EDI`	Dissociative Ionization	1660	1660
`EDR`	Dissociative Recombination	88	88
`EDS`	Dissociation	110	110
`EDT`	Electron Detachment	1	0
`EDX`	De-Excitation	3	3
`EEL`	Elastic Scattering	24	0
`EEX`	Excitation	608	0
`EIN`	Ionization	1964	1199
`EMI`	Multiple Ionization	1	0
`EMT`	Momentum Transfer	5	0
`ERR`	Radiative Recombination	1053	0
`ETS`	Total Scattering	1	0
`EXE`	Electronic Excitation	63,724	63,724
`EXV`	Vibrational Excitation	67,395	67,395
`HAC`	Association	15	0
`HCX`	Charge Transfer	27,812	730
`HDC`	Dissociative Charge Transfer	85	86
`HDE`	Dissociative Excitation	10	10
`HDI`	Dissociative Ionization	39	39
`HDS`	Dissociation	72	73
`HDT`	Detachment	16	6
`HDX`	De-Excitation	52	0
`HES`	Elastic Scattering	76	51
`HEX`	Excitation	565	12
`HHT`	Transport	82	52
`HIN`	Ionization	289	74
`HIR`	Interchange Reactions	23	36
`HMI`	Multiple Ionization	24	5
`HMN`	Mutual Ion-Ion Neutralization	4	0
`HMT`	Momentum Transfer	82	52
`HPN`	Penning Ionization	2	0
`HST`	Electron Stripping	83	17
`HTI`	Transfer Ionization	2	2
`HXE`	Electronic Excitation	3	3
`HXV`	Vibrational Excitation	586	586
`PED`	Elastic Diffusion	239	0
`PES`	Elastic Scattering	239	0
`PEX`	Photoexcitation	239	0
`PIN`	Photoionization	7725	0
`PRD`	Radiative Decay	600	600

3. Materials and Methods

3.1. Data Model

The CollisionDB database has been created to store and manage large amounts of atomic and molecular collisional data. It is built on a relational database management system backend: the main tables and relationships are described in the extended Entity Relationship Diagram (EER) shown in Figure 7. Here, we provide an overview of the main tables and their relationships, along with a description of the basic metadata stored in each table.

Each collisional dataset in the database is described by the primary table `cds_cdataset`, which includes the following metadata attributes:

- `id`: a unique integer primary key identifying the dataset within the CollisionDB ecosystem.
- `reaction_id`: a foreign key identifying the single collisional process with which a dataset is associated (represented in the `rxn_reaction` table seen below).
- `method`: the method used to obtain the numerical data: one of `"experiment"`, `"theory"`, `"semi-empirical"` or `"estimate"`. More precise details about the computational method used to calculate data can be specified using pre-defined abbreviations in the `json_data` attribute (see below).
- `data_type`: the type of collisional data, which can be one of the following: `"cross section"`, `"differential cross section"` or `"rate coefficient"`.
- `comment`: a free-text comment field providing additional information concerning the dataset.
- `json_data`: all other metadata such as threshold, uncertainty and units, stored as key–value pairs in a JSON (JavaScript Object Notation) object (see Section 3.2).
- `recommended`: A Boolean flag, indicating whether the dataset is evaluated and recommended (details may be provided in the `comment` field).
- `deprecated`: A Boolean flag, indicating whether the dataset is out of date due to identified errors or quality issues. Details can be provided in the `deprecation_comment` field. A new dataset may be available in the database.
- `time_added`, `time_deprecated`: timestamps indicating when a dataset was added or deprecated in the CollisionDB database.
- One or more bibliographic references for the data, identified through a many-to-many relationship with the `refs_ref` table.

This schema ensures that each dataset has appropriate and unambiguous provenance and contextual metadata to ensure its accurate use. It will often be the case that a single collisional process is associated with multiple datasets. For instance, a given reaction can be described by different methods, such as an experiment, theory or a different computational approach, or have a different energy range or data type, such as a cross section or rate coefficient.

The `rxn_reaction` table describes each collisional process ("reaction") through the structure outlined below; each `cds_cdataset` dataset entry is associated with exactly one `rxn_reaction` entry.

- `text`: a text representation of the reaction in a canonical form conforming to the standards of and parseable by the `PyValem` library [8].
- `html`, `latex`: HTML and LaTeX representations of the reaction, for display in the browser and export.
- `comment`: a free-text comment field providing further information about the reaction.
- `ordered_text`: a text representation of the reaction in which the reactants and products are ordered in an arbitrary but consistent way to facilitate indexing, searching and comparison of reactions.
- `rxn_reaction_reactants` and `rxn_reaction_products`: these tables provide a many-to-many relationship between each reaction and its individual reactant and product species (including their quantum states, where relevant), which are held in the `rp_rp` table.
- `rxn_reaction_process_types`: this table provides a many-to-many relationship between a reaction and its classifying process codes (e.g., `EIN` = electron-impact ionization); a complete list of these codes is given in Ref. [9].

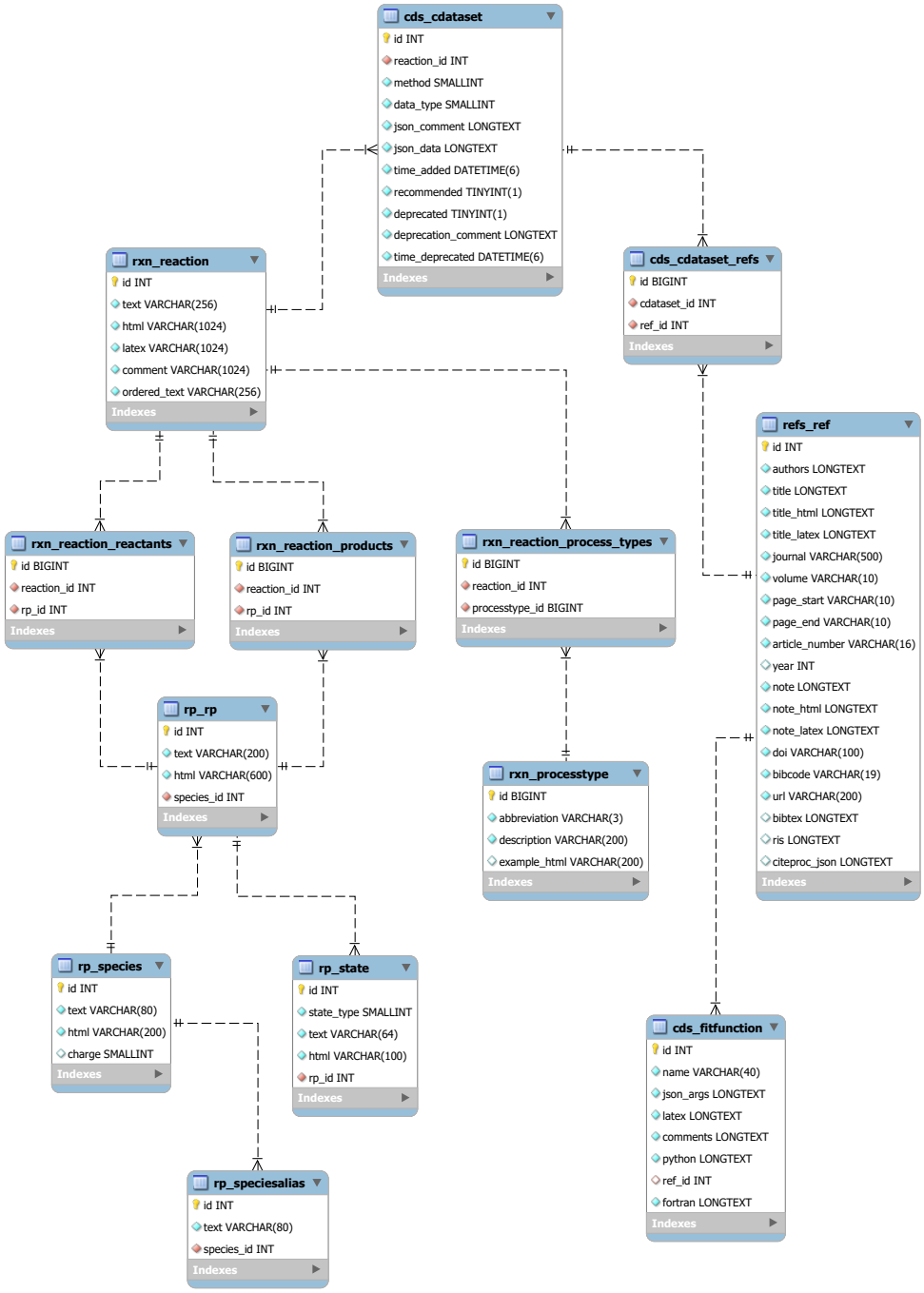

Figure 7. EER diagram of the CollisionDB relational database schema.

Reactants and products are stored in the `rp_rp` table in plain-text format conforming to the standardized syntax implemented by the PyValem library. They can be "stateful" in the sense that, in addition to identifying the chemical species, they can contain information about any relevant quantum numbers, labels and symmetries. Species themselves can be atoms, molecules, ions or isotopes, including the electron (`e-`) and the photon (`hν`).

To aid with the search functionality, these stateful species can have one or more aliases (stored in the table `rp_speciesalias`). For example, the isotopologue of molecular hydrogen usually written as `HD` has aliases `DH`, `H(2H)`, `(2H)H`, `(1H)(2H)` and `(2H)(1H)`. Aliases can also be other chemical identifiers, for example, the InChIKey, and quantum

states can be specified as atomic or molecular electronic configurations, term symbols and individual quantum numbers in key–value pairs. Some examples are given in Table 1.

CollisionDB also supports analytic fits to collisional data where available. Metadata for fitting functions can be stored in the `cds_fitfunction` table, including the arguments of the fitting function, their descriptions, types and units in the `json_args` attribute. CollisionDB also provides implementations (source code) of the fitting functions themselves in the Python programming language.

A description of the metadata formats and standards used to populate this database is given in the following subsections.

3.2. Data Transfer Format for Download: JSON

Each dataset in CollisionDB is described by metadata which are stored in the database and also presented as a header, in JSON format, to each data file. The meaning of the JSON metadata objects is given in Table 2, and an example metadata object is given in Listing 1. The data are separated from the metadata header by a single line of at least five hyphens.

Uncertainties in the numerical data may be given as a global estimate applying to all data points (using the `unc_perc` key) or per-data point; in the latter case, the uncertainty is separated from the data value by a colon and may be either a symmetric range `v:u` implying $v \pm u = v^{+u}_{-u}$ or asymmetric `v:l:u` implying v^{+u}_{-l}.

3.3. Data Transfer Format for Upload

3.3.1. Plain Text

For the convenience of data providers, data can be submitted to the CollisionDB database in UTF-8 encoded, plain-text format; an example template file is given in Listing 10. The file includes metadata specified as `key=value` pairs, followed by the numerical data in space-delimited columns. The metadata keys are described in Table 2 and online at https://amdis.iaea.org/db/collisiondb/submitting-data/ (accessed on 23 March 2024). Their values may be numbers, strings (inside double quotes, `"..."`) or lists (a comma-separated sequence of values, enclosed in square brackets, `[...]`, depending on the key. Blank lines and any content following the "`#`" symbol are ignored; missing data values are given by asterisks.

Listing 10. An example template file for submission.

```
reaction="e- + HD X(1SIGMA+g); v=0 -> e- + HD C(1PIu); v=0"
process_types=["EXE","EXV"]

data_type="cross section"
method="MCCC"
threshold="12.3259"
columns=["E, eV", "sigma, m2"]
comment="MCCC calculations of vibrationally-resolved electronic excitation of HD, adiabatic
      nuclei calculations performed with the spheroidal MCCC(210) model"
doi=["10.1016/j.adt.2020.101403"]

E             sigma
1.250E+01   3.05E-24                        # Example without uncertainty value
1.550E+01   *                               # Example of a missing data point
1.229E+02   3.05E-22:3.05E-23               # Example of symmetric uncertainty
1.239E+02   2.974E-22:2.99E-23,5.80E-23     # Example of an asymmetric uncertainty
```

3.3.2. JSON

Data are uploaded to the CollisionDB database from a JSON object, and data providers can also choose to submit their data directly in JSON format, as shown in Listing 11. This example represents the JSON object containing metadata and numerical data of a collisional dataset, as specified in Listing 10 in the plain-text format. Numerical data undergo standardized unit conversions using the the PyQn Library [10] as well as numerical checks. These verify, for example, that the grid of energies or temperatures is monotonically increasing.

This JSON object is processed into a well-defined format conforming to the standardized schemas established by the IAEA's AMD Unit and other experts and validated prior to uploading it to the database. This involves the use of the PyValem Library [8] to canonicalize reactions and validate charge balance and stoichiometry conservation, etc. Additionally, the DOIs (Digital Object Identifiers) are resolved into the citable format using the Python package django-pyref [17]. In order to prevent duplication, the import script checks for pre-existing metadata in CollisionDB.

Although the metadata are uploaded as `cdataset` instances into the relational database, numerical data (including the JSON metadata header) are saved as static files.

Listing 11. An example JSON dataset file for submission.

```
{
    "listing2.txt": {
        "metadata": {
            "reaction": "e- + HD X(1SIGMA+g); v=0 -> e- + HD C(1PIu); v=0",
            "process_types": ["EXE", "EXV"],
            "data_type": "cross section",
            "method": "MCCC",
            "threshold": "12.326",
            "columns": [
                {
                    "name": "E",
                    "units": "eV"
                },
                {
                    "name": "sigma",
                    "units": "cm2"
                }
            ],
            "comment": "MCCC calculations of vibrationally-resolved electronic excitation
                 of HD, adiabatic nuclei calculations performed with the spheroidal MCCC
                 (210) model",
            "doi": ["10.1016/j.adt.2020.101403"]
        },
        "data": {
            "e- + HD X(1SIGMA+g); v=0 -> e- + HD C(1PIu); v=0": {
                "1.250e+01": {
                    "sigma": 3.05e-20
                },
                "1.229e+02": {
                    "sigma": 3.05e-18,
                    "unc": 3.05e-19
                },
                "1.239e+02": {
                    "sigma": 2.974e-18,
                    "unclo": 2.99e-19,
                    "unchi": 5.80e-19
                }
            }
        }
    }
}
```

4. Conclusions

In providing easy access to peer-reviewed published data, conforming to FAIR principles and being facilitated by structured metadata and API integration, it is hoped that CollisionDB can provide a useful resource to the plasma collisional physics community and open up new possibilities for machine-learning-driven advances in fusion and other areas of plasma research.

Author Contributions: C.H. conceived the idea and the initial schema of the CollisionDB database. D. and C.H. wrote the manuscript and managed data uploading and database maintenance. C.H., D. and M.H. contributed to software development. All authors reviewed the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The first release version of all datasets, v2023.1, is available on the CollisionDB website at https://amdis.iaea.org/db/collisiondb/ (accessed on 23 March 2024) under the terms of the CC BY 4.0 license. The PyCollisionDB Python library for interacting with the database is available at https://github.com/xnx/pycollisiondb/ (accessed on 23 March 2024) and is released under the terms of Apache license.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Pitchford, L.C.; Alves, L.L.; Bartschat, K.; Biagi, S.F.; Bordage, M.-C.; Bray, I.; Brion, C.E.; Brunger, M.J.; Campbell, L.; Chachereau, A.; et al. LXCat: An Open-Access, Web-Based Platform for Data Needed for modeling low temperature plasmas. *Plasma Process. Polym.* **2017**, *14*, 1600098. [CrossRef]
2. Park, J.H.; Choi, H.; Chang, W.S.; Chung, S.Y.; Kwon, D.C.; Song, M.Y.; Yoon, J.S. A New Version of the Plasma Database for Plasma Physics in the Data Center for Plasma Properties. *Appl. Sci. Converg. Technol.* **2020**, *29*, 5–9. [CrossRef]
3. Celiberto, R.; Armenise, I.; Cacciatore, M.; Capitelli, M.; Esposito, F.; Gamallo, P.; Janev, R.K.; Laganà, A.; Laporta, V.; Laricchiuta, A.; et al. Atomic and molecular data for spacecraft re-entry plasmas. *Plasma Sources Sci. Technol.* **2016**, *25*, 033004. [CrossRef]
4. Tennyson, J.; Mohr, S.; Hanicinec, M.; Dzarasova, A.; Smith, C.; Waddington, S.; Liu, B.; Alves, L.L.; Bartschat, K.; Bogaerts, A.; et al. The 2021 release of the Quantemol database (QDB) of plasma chemistries and reactions. *Plasma Sources Sci. Technol.* **2022**, *31*, 095020. [CrossRef]
5. Summers, H.P. The ADAS User Manual, Version 2.6.2004. Available online: http://www.adas.ac.uk (accessed on 23 March 2024).
6. Hulse, R.A. The ALADDIN atomic physics database system. *AIP Conf. Proc.* **1990**, *206*, 63–72. [CrossRef]
7. Wilkinson, M.D.; Dumontier, M.; Aalbersberg, I.J.; Appleton, G.; Axton, M.; Baak, A.; Blomberg, N.; Boiten, J.-W.; da Silva Santos, L.B.; Bourne, P.E.; et al. The FAIR Guiding Principles for scientific data management and stewardship. *Sci. Data* **2016**, *3*, 160018. [CrossRef] [PubMed]
8. Hill, C. PyValem, GitHub Repository. 2022. Available online: https://github.com/xnx/pyvalem (accessed on 23 March 2024).
9. Hill, C.; Dubernet, M.L.; Endres, C.; Karwasz, G.; Marinković, B.; Marquart, T.; Heinola, K.; Zwölf, C.M.; Moreau, N.; Dipti; et al. "Classification of Processes in Plasma Physics" Version 2.4. 2022. Available online: https://amdis.iaea.org/media/miscellaneous-publications/plasma-processes-classification-v2.4.pdf (accessed on 23 March 2024).
10. Hill, C. PyQn, GitHub Repository. 2022. Available online: https://github.com/xnx/pyqn (accessed on 23 March 2024).
11. Hill, C. PyCollisionDB, GitHub Repository. 2022. Available online: https://github.com/xnx/pycollisiondb (accessed on 23 March 2024).
12. Shah, M.B.; Gilbody, H.B. Experimental study of the ionisation of atomic hydrogen by fast H^+ and He^{2+} ions. *J. Phys. B At. Mol. Opt. Phys.* **1981**, *14*, 2361. [CrossRef]
13. Shah, M.B.; Elliott, D.S.; Gilbody, H.B. Ionisation of atomic hydrogen by 9–75 keV protons. *J. Phys. B At. Mol. Opt. Phys.* **1987**, *20*, 2481. [CrossRef]
14. Agueny, H.; Hansen, J.P.; Dubois, A.; Makhoute, A.; Taoutioui, A.; Sisourat, N. Electron capture, ionization and excitation cross sections for keV collisions between fully stripped ions and atomic hydrogen in ground and excited states. *At. Data Nucl. Data Tables* **2019**, *129–130*, 101281. [CrossRef]
15. Leung, A.C.K.; Kirchner, T. Proton impact on ground and excited states of atomic hydrogen. *Eur. Phys. J. D* **2019**, *73*, 246. [CrossRef]
16. Hill, C.; Dipti; Heinola, K.; Dubois, A.; Sisourat, N.; Taoutioui, A.; Agueny, H.; Tőkési, K.; Ziaeian, I.; Illescas, C.; et al. Atomic collisional data for neutral beam modeling in fusion plasmas. *Nucl. Fusion* **2023**, *63*, 125001. [CrossRef]
17. Hill, C. django-pyref, GitHub Repository. 2022. Available online: https://github.com/xnx/django-pyref (accessed on 23 March 2024).