EPN-TAP: Publishing Solar System Data to the Virtual Observatory
Stéphane Erard, Baptiste Cecconi, Pierre Le Sidaner, Markus Demleitner, Mark Taylor

To cite this version:

HAL Id: hal-03999093
https://hal.science/hal-03999093
Submitted on 21 Feb 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers. L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
EPN-TAP: Publishing Solar System Data to the Virtual Observatory

Version 2.0

IVOA Recommendation 2022-08-22

Working Group
Data Access Layer

This version
https://www.ivoa.net/documents/EPNTAP/20220822

Latest version
https://www.ivoa.net/documents/EPNTAP

Previous versions
This is the first public release

Author(s)
Stéphane Erard, Baptiste Cecconi, Pierre Le Sidaner, Markus Demleitner, Mark Taylor

Editor(s)
Stéphane Erard, Baptiste Cecconi

Abstract

This document defines the EPN-TAP framework, which is using TAP with the EPNCore metadata dictionary. The EPNCore metadata dictionary defines the core components that are necessary to perform data discovery in the Solar System related science fields. It includes parameters to describe data products coverage (temporal, spectral, spatial, photometric), origin (instrument, facility), content (target, physical parameters), access, references, etc. Its implementation with TAP (Table Access Protocol) is presented, including service registration guidelines. Topical extension metadata dictionaries are also presented.
Status of this document

This document has been reviewed by IVOA Members and other interested parties, and has been endorsed by the IVOA Executive Committee as an IVOA Recommendation. It is a stable document and may be used as reference material or cited as a normative reference from another document. IVOA’s role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability inside the Astronomical Community.

A list of current IVOA Recommendations and other technical documents can be found at https://www.ivoa.net/documents/.

Contents

1 Introduction 4
   1.1 EPN-TAP design ....................... 5
   1.2 EPN-TAP rules applying to table structure and parameter usage ........... 6
   1.3 Role within the VO Architecture ........................................... 8

2 The EPNCore metadata parameter dictionary 9
   2.1 Mandatory parameters ............................................................. 9
       2.1.1 Granule references ....................................................... 9
       2.1.2 Data Description ............................................................. 10
       2.1.3 Target description .......................................................... 13
       2.1.4 Axes .............................................................................. 16
       2.1.5 Data origin ........................................................................ 23
       2.1.6 Granule call-back info ....................................................... 24
   2.2 Optional parameters ................................................................. 25
       2.2.1 Data Access Reference ....................................................... 25
       2.2.2 Miscellaneous file metadata ............................................... 26
       2.2.3 Supplementary description ............................................... 27
       2.2.4 Description of coordinate frame ......................................... 29
       2.2.5 Target configuration and observing geometry .......................... 30
       2.2.6 Vertical scales on planets ................................................... 32
   2.3 Extensions ................................................................................. 32
       2.3.1 Particle spectroscopy extension ....................................... 32
       2.3.2 Solar System Objects extension ......................................... 33
       2.3.3 Maps ............................................................................... 34
       2.3.4 Contributive works ............................................................ 34
       2.3.5 Experimental spectroscopy .................................................. 35
       2.3.6 APIS extension .................................................................... 38
       2.3.7 Events .............................................................................. 38
       2.3.8 Other extensions ................................................................. 39
Acknowledgments

EPNCore has been developed by the VESPA (Virtual European Solar and Planetary Access) team. VESPA has been designed in the frame of Europlanet 2020 Research Infrastructure project and finalized during the Europlanet 2024 Research Infrastructure project, based on an assessment phase in the Europlanet Research Infrastructure programme, JRA4 work package (IDIS activity).

The Europlanet-RI project was funded by the European Commission under the 7th Framework Program, grant 228319 “Capacities Specific Programme”. The Europlanet 2020 Research Infrastructure project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 654208. The Europlanet-2024 Research Infrastructure project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 871149. Additional funding was provided in France by the CNRS / Action Spécifique Observatoire Virtuel and Programme National de Planétologie / INSU, as well as CNES, through the participation to IPDA activities.

Conformance-related definitions

The words “MUST”, “SHALL”, “SHOULD”, “MAY”, “RECOMMENDED”, and “OPTIONAL” (in upper or lower case) used in this document are to be interpreted as described in IETF standard RFC2119 (Bradner, 1997).

The Virtual Observatory (VO) is a general term for a collection of federated resources that can be used to conduct astronomical research, education, and outreach. The International Virtual Observatory Alliance (IVOA) is a global collaboration of separately funded projects to develop standards and infrastructure that enable VO applications.

List of Acronyms

- ADQL Astronomical Data Query Language
- APIS Auroral Planetary Imaging and Spectroscopy
1 Introduction

This document defines the EPN-TAP framework. EPN-TAP is a protocol used to describe and access data related to the study of the Solar System in general, including observational, modeled, and experimental data. It consists of 1) the usual TAP mechanism; 2) the EPNCore metadata dictionary; 3) a set of rules defining table structure and column usage. EPN-TAP relies on the TAP (Table Access Protocol) standard with no modification (Dowler and Rixon et al., 2019), so that only items 2 and 3 are described here. All EPN-TAP services are accessible through standard TAP clients.

Its implementation with TAP is presented, including parameter usage (sections 2 and 3) and service registration guidelines (section 4).

The first version of EPN-TAP was developed as an assessment study during the Europlanet-RI programme (2011), and was limited to test purposes. It was redesigned to EPN-TAP 2.0 before the start of Europlanet 2020 RI and completed during the following programme Europlanet 2024 RI based on a large set of data services and use cases. This later version is described here.
1.1 EPN-TAP design

The EPNCore metadata dictionary defines the core components that are necessary to perform data discovery in science fields related to the Solar System and related fields. The EPNCore metadata dictionary includes parameters to describe data products coverage (temporal, spectral, spatial, illumination conditions), origin (instrument, facility), content (target, physical parameters), access, references, etc. The notion of parameter is derived from the “keywords” of the PDS standard used to archive planetary space missions. They are intended either as search parameters or descriptive ones, although in TAP they can all be searched by value. In this sense, EPNCore bears similarities with the ObsCore standard (Louys and Tody et al., 2017) from which it borrows several concepts.

EPN-TAP also uses the notion of granule (inherited from the SPASE standard) to refer to the data service granularity, which is smallest data unit provided by a service and described individually in the associated table. A “granule” can correspond to a data file, a set of scalar values, a call to a web service, a query to a data service in a different protocol, etc. Grouping of data into granules is at the discretion of the data provider. This may be equivalent to “dataset” or “data product” in other contexts, but these words are avoided here because they have a different meaning in many Planetary Science archives.

An EPN-TAP service therefore consists of a single table describing a list of granules with the metadata dictionary: a granule corresponds to a row of the table, while a parameter corresponds to a column. Only one data product can be described and linked per row of the table. Observations, simulations, or experimental data are supported. Several independent services can reside on a single server.

The global philosophy of EPN-TAP is to describe all tables with a common set of mandatory parameters which can be used to query all EPN-TAP services together, which is the scope of the VESPA portal. A unique query can then be sent to and answered by all data services without generating errors. This also requires that all parameter values are provided in a standard form, in particular with standard units. In other words, mandatory parameters have to be present in an EPN-TAP service and must provide values in standard form. Although most of them can be left empty when non-relevant or unknown, care should be taken to fill as many parameters as possible to provide a better service to the user.

In addition to these mandatory parameters, the EPN-TAP dictionary includes optional parameters which are grouped in two levels: 1) common, general purpose parameters provide complementary information; 2) parameters from thematic extensions have been defined from a set of related data services and must be used consistently in data services in these fields. The first group includes file access information and additional description of data and target. The second group defines additional axes and provides extra parameters to describe similar observations consistently. The optional parameters can be used independently of each other (except the min/max pairs which need to appear together).

Finally, EPN-TAP allows data providers to use entirely new parameters in a data service to provide specific information, when no existing parameter applies. This may occur in particular when the data consist in a few scalar values included in the table itself as individual parameters. Such parameters are best defined in consistency with existing ones, and may eventually be
included in thematic extensions. Parameter names are given in lower case, they must not include spaces or special characters; underscore is used as separator.

Section 2 provides a description of parameters and their usage. All the currently defined parameters are listed in Section 3 (table) with their type, unit, and UCD.

EPNCore usage is not limited to data distribution and access through TAP. It also provides a standard way to describe Planetary Science data, e.g., to handle local databases and private projects while benefiting from interfaces with VO tools, in particular with TOPCAT, Aladin, and CASSIS, which were closely associated to this development.

1.2 EPN-TAP rules applying to table structure and parameter usage

• Rules related to data services and table structure include:
  
  – A service consists of one database schema containing only one table. The table must be called <schema>.epn_core.
  
  – Only one data product is linked per table row, and parameters in that row describe that product. In case the same data product is distributed with several formats, there can be either one row per format, or a single row providing access to the preferred format, while other formats are distributed through an associated datalink.
  
  – Related files providing documentation of the data product can however be associated through thumbnail_url (quick-look image), datalink_url (various documents and links), and external_link (web page providing more discussion of this granule).
  
  – Data services are declared in the IVOA registry of services according to TAP guidelines (section 4).
  
  – The service metadata must include the usual elements to allow discovery in the registry. Subject elements should be taken from the IVOA flavored Unified Astronomy Thesaurus (UAT) when available. At least one of these top-level keywords of the UAT must be provided: exoplanet-astronomy, solar-physics, solar-system-astronomy, planetary-science.
  
  – Data service output VOTables should include TIMESYS and COOSYS elements when constant. This is not always true, see time_scale and space_origin parameters for details.
  
  – Data service output must pass validation; an EPN-TAP service validator is included in STILTS/taplint\(^1\).

• Rules related to parameters include:
  
  – Parameters defining a range along an axis most often appear as a pair with *_min and *_max variations. Both values must be provided to support ranges correctly. If only one value is available, it must appear in both parameters.

\(^1\)http://www.starlink.ac.uk/stilts/sun256/taplint.html
Parameters accept values in lower case, except when a global standard or vocabulary applies. This is the case in particular with Solar System object names, which follow the IAU naming convention.

Empty parameters can be void or can contain the NULL value.

Most floating-point parameters only require single precision, with the exception of time_min and time_max which must be stored and returned in double precision to preserve the required accuracy.

Some string parameters can be multivalued. Lists of values must be provided as hash-separated lists (values separated by internal # character).

Although each row contains at most one linked data product, special conventions exist: Thumbnails may be provided on the same granule row, as far as they are intended for quick-look in the VESPA portal or similar user interfaces; inclusion of thumbnails is recommended. In contrast, large previews should be handled as separated data products. Associated documents, or in some cases alternative formats, can be attached via the datalink_url parameter. Related web services or SODA data services are also linked via this parameter.

Some parameters only accept values from a predefined list. Such lists are provided or discussed in section 2 and are maintained on an external web page with a Permanent ID: https://hdl.handle.net/21.15110/epn_tap_extensions. Whenever possible, this refers to IVOA vocabularies maintained independently.

Internal spaces are supported in string parameters values, but not leading or trailing spaces.

Special characters, quotes, and hash are not allowed in string parameter values, and may be changed to _ (underscore, which is the single-character wildcard in ADQL LIKE conditions).

Although additional, service-specific parameters can be used, care must be taken to avoid duplications or variations on existing parameters, including those from thematic extensions.

Such free parameters, when very specific to a service, may use a prefix related to this service to prevent conflicts (e.g., myservice_myparameter). Associated UCDs must be extracted from the current version of the standard (UCD1+).

When free parameters contain a numerical value, associated errors should use the syntax: parameter_error, or parameter_error_min and parameter_error_max when asymmetrical (all non-negative). The associated UCDs must start with “stat.error;”, “stat.error;stat.min;”, or “stat.error;stat.max;”, respectively.

Upper and lower limits can be introduced with the same 3 parameters used to provide asymmetrical errors, say: val, val_error_min and val_error_max. The following scheme is suggested:

An upper limit can be provided with val = 0, val_error_min = Inf, val_error_max = upper_limit.
- A lower limit can be provided with \( \text{val} = \text{lower\_limit}, \text{val\_error\_min} = 0, \text{val\_error\_max} = \text{Inf} \).

This scheme minimizes interpretation errors, simplifies graphical representations, and preserves transforms such as value \( \rightarrow \) -value.

### 1.3 Role within the VO Architecture

![Architecture diagram for this document](image)

*Figure 1: Architecture diagram for this document*

Figure 1 shows how EPN-TAP fits into Virtual Observatory architecture. EPN-TAP’s main connections to other VO standards are:

**ADQL** (Osuna and Ortiz et al., 2008)
ADQL is the preferred language to query EPN-TAP in. The common language is a prerequisite for global discovery within all EPN-TAP services.

**TAP** (Dowler and Rixon et al., 2019)
TAP is used to transport queries, table metadata, and the results of queries. The re-use of TAP gives EPN-TAP a wide range of client and server implementations.
VODataService (Demleitner and Plante et al., 2021)
VODataService tableSets are used to discover EPN-TAP services in VO Registries.

UCDs (Cecconi and Louys et al., 2021)
UCDs associated to EPNCore parameters provide context when manipulating EPN-TAP tables and result VOTables. They also describe the physical quantities distributed in services via the measurement_type parameter, making searches by physical quantity possible.

DALI (Dowler and Demleitner et al., 2017)
DALI lays out several basic conventions, including the serialization of data types like timestamps and geometries.

2 The EPNCore metadata parameter dictionary

In EPN-TAP services, EPNCore parameters appear as table columns and describe the granules. Mandatory parameters must be present in all EPN-TAP services, while optional parameters can be included as needed, with specified units and UCD.

2.1 Mandatory parameters

2.1.1 Granule references

These 3 parameters must always be provided and contain a value. If granule grouping is not relevant, use a single value for granule_gid (and obs_id can be identical to granule_uid).

**granule_uid** Identifier for this row. This parameter is a primary key in epn_core tables, i.e., no two different rows may share the same granule_uid.

There can be only one data file associated to a granule (plus possibly a thumbnail for quick-look purpose in a search interface).

Basic ascii characters are allowed, including internal spaces, except for the # character.

**obs_id** Associates granules derived from the same data (e.g., various representations or processing levels). May be the ID of the original observation.

**granule_gid** Associates granules of same type (e.g., same map projection, or geometry data products) — think of it as a simple and convenient way to group or differentiate types of data.

When several files relate to the same data, this parameter helps distinguishing them: this will allow the user to select the type of data of interest. For instance, a service may provide links to calibrated images, plus raw data and ancillary information for every observation; these will share the same obs_id, but will have different granule_gid.
2.1.2 Data Description

data_product_type  The data_product_type parameter describes the high level scientific organization of the data product linked by the access_url parameter, or directly included in the table (in which case the value is ‘ci’ for catalogue_item). EPNCore currently defines several types listed below. Data providers must select the type most adapted to their data. In complex situations (e.g., when a file contains several data products), several types can be used to describe the same granule by using a hash-separated-list — although using several granules to describe the file content may be a better solution.

In EPN-TAP these types are identified by a 2-characters ID, so that multivalued queries are unambiguous. Possible IDs are listed below with their meaning:

- **im** (= image): scalar field with two spatial axes, or association of several such fields, e.g., images with multiple color planes, from multichannel or filter cameras. Preview images (e.g., map with axis and caption) also belong here. Conversely, vectorial 2D fields are described as spatial_vector (see below).

- **ma** (= map): scalar field / rasters with two spatial axes covering a large area and projected either on the sky or on a planetary body, associated to spatial_coordinate_description and map_projection parameters (with a short enumerated list of possible values); each pixel is associated to 2D coordinates (e.g., fits with WCS). This is mostly intended to identify radiometrically calibrated and orthorectified images with complete coverage that can be used as reference basemaps, but this also includes HiPS.

- **sp** (= spectrum): measurements organized primarily along a spectral axis, e.g., radiance spectra. This includes spectral aggregates (series of related spectral segments with non-connected spectral ranges, e.g., from several channels of the same instrument, various orders from an échelle spectrometer, composite spectra, SED, etc).

- **ds** (= dynamic_spectrum): consecutive spectral measurements through time, organized primarily as a time series. This typically implies successive spectra of the same target / field of view.

- **sc** (= spectral_cube): sets of consecutive spectral measurements with 1 or 2D spatial coverage, e.g., imaging spectroscopy. The choice between image and spectral_cube is dictated by the characteristics of the instrument (which dimension is most resolved and which dimensions are acquired simultaneously). The choice between dynamic_spectrum and spectral_cube is related to the uniformity of the field of view and by practices in the science field.

- **pr** (= profile): scalar or vectorial measurements along 1 spatial dimension, e.g., atmospheric profiles, atmospheric paths, sub-surface profiles, traverses...

- **pf** (= photometric_function): scalar or vectorial measurements along 1 angular dimension, e.g., phase or polarization curves, phase functions, emission-phase function sequences... Does not handle variations along several angular axes. This is typically associated to variations in illumination angle parameters.
• **vo** (= volume): measurements with 3 spatial dimensions, e.g., internal or atmospheric structures, including shells/shape models (3D surfaces).

• **mo** (= movie): sets of chronological 2D spatial measurements (consecutive images)

• **cu** (= cube): multidimensional data with 3 or more axes, e.g., all that is not described by other 3D data types such as spectral cube, volume, or movie. This is intended to accommodate unusual data with multiple dimensions. This can be used for 3D ancillary data associated to spectral cubes, e.g., providing the coordinates or illumination angles for each spectrum.

• **ts** (= time_series): measurements organized primarily as a function of time (with exception of dynamical spectra and movies, i.e., usually a scalar quantity). Typical examples of time series include space-borne dust detector measurements, daily or seasonal curves measured at a given location (e.g., a lander), and light curves.

• **ca** (= catalogue): applies to a granule providing a catalogue of object parameters, a list of features, a table of granules in another TAP service, a list of events, a list of spectral lines... The result metadata table of a service query can also be considered as a catalogue. Catalogues can be provided as VOTable (possibly containing multiple tables, although this is not supported by SAMP). It is good practice to describe the type of data included in the catalogue using a hash-separated-list (e.g., a table of spectra should be described by ca#sp, so that it will respond to a query for spectra).

• **ci** (= catalogue_item): applies when the service itself provides a catalogue with entries described as individual granules, in particular when there is no associated file (e.g., a list of asteroid properties or spectral lines). Catalogue_item can be limited to scalar quantities (including strings), and possibly to a single element. This organization allows the user to search inside the catalogue from the TAP query interface. In practice, Spice kernels are identified as catalogue_items because they are usually associated to a set of scalar parameters.

• **sv** (= spatial_vector): vector information associated to localization, such as a spatial footprints, a GIS-related element, etc — e.g., a kml or geojson file (STC-S strings are provided though the s_region parameter, though). This includes maps of vectors, e.g., wind maps.

• **ev** (= event): introduces individual VOevents formatted according to IVOA standard (or possibly events with other formatting). Characteristics are provided via the event_* parameters.

**Example TAP queries:**

```
SELECT TOP 1000 * FROM iks.epn_core
WHERE dataproduct_type LIKE '%sp%'
```
 measurement_type  The measurement_type parameter defines the physical quantities contained in the data, using UCDs (Cecconi and Louys et al., 2021). It relates to the reported quantity, not to the type of experiment. Therefore, only UCDs related to physical quantities can be used; e.g., phys.absorption;em.opt.I is eligible, while pos.lunar.occult (occultation by lunar limb) is not. This is used in particular to provide indications to visualization/processing tools.

Whenever several quantities are comprised in the data, the measurement_type parameter must describe all these quantities, using multiple UCDs in a hash-separated-list.

**Example:**

- For images in general (i.e., actual measurements with a camera), the relevant UCD is phot.radiance;obs.image (or phot.radiance;obs.image;stat.uncalib if not calibrated).

- For spectra: phot.radiance describes radiance, phys.reflectance stands for reflectance factor (= I/F), and phot.flux.density for a flux vector (irradiance). The associated spectral vector is described by UCDs em.wl, em.freq or em.energy, and the related error has the form stat.error;phys.reflectance (for I/F).

- Quantities derived from modeling or simulations are described by the regular UCD with “;meta.modelled” appended.

processing_level  The processing_level parameter is intended to provide the user with a quick evaluation of data readiness level. EPN-TAP uses a simplified scheme as described below.

Several classifications are in use in different contexts, as summarized in Table 2.1.2. EPNCore uses the CODMAC / PDS3 levels but removes the intermediate calibration levels; this is equivalent to using the simplified PDS4 levels and maintaining a separate level for ancillary data. “Partially calibrated” data collections are in general considered as not calibrated, but this evaluation is up to the data provider, depending on context. “Ancillary” data include all extra information documenting the measurements, in particular coordinates or geometry files. Several processing levels can be included in the same service (in particular calibrated and ancillary data, but also raw data). When mixed in the same file, the choice is left to the data provider.

Note that the optional processing_level_desc parameter is available to provide, e.g., a specific encoding of processing levels related to a data collection, or more details about partial calibrations.

Most EPN-TAP data services are expected to include Calibrated or Derived data.
<table>
<thead>
<tr>
<th>EPN-TAP2</th>
<th>CODMAC</th>
<th>PSA</th>
<th>NASA</th>
<th>PDS3</th>
<th>PDS4</th>
<th>Obs/TAP</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 (raw)</td>
<td>(0?)</td>
<td>UDR</td>
<td>Telemetry</td>
<td>0</td>
<td>Unprocessed Data Record (low-level encoding, e.g., telemetry from a spacecraft instrument. Normally available only to the original team)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 (edited)</td>
<td>1</td>
<td>0</td>
<td>EDR</td>
<td>Raw</td>
<td>1</td>
<td>Experiment Data Record (often referred to as “raw data”: decommutated, but still affected by instrumental effects)</td>
</tr>
<tr>
<td>2 or 3</td>
<td>_</td>
<td>_</td>
<td>_</td>
<td>_</td>
<td>Partially calibrated</td>
<td>Processed beyond the raw stage, but have not yet reached calibrated status (PDS4)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3 (calibrated)</td>
<td>2</td>
<td>1A</td>
<td>RDR</td>
<td>Calibrated</td>
<td>2</td>
<td>Reduced Data Record (“calibrated” in physical units, no resampling)</td>
</tr>
<tr>
<td>5 (yes, 5)</td>
<td>4 (resampled)</td>
<td>1B</td>
<td>REFDR</td>
<td>Derived</td>
<td>3</td>
<td>Reformatted Data Record (mosaics or composite of several observing sessions, involving some level of data fusion)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5 (derived)</td>
<td>3</td>
<td>2-5</td>
<td>DDR</td>
<td>Derived</td>
<td>4</td>
<td>Derived Data Record (result of data analysis, directly usable by other communities with no further processing)</td>
</tr>
<tr>
<td>6</td>
<td>6 (ancillary)</td>
<td>ANCDR</td>
<td>Derived</td>
<td>Ancillary Data Record (extra data specifically supporting a data set, such as coordinates, geometry... but also dark currents, flat fields...)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes:

- This table is a compilation of information from PSA, PDS4, and ObsCore documents
- The PDS3 column corresponds to the PDS3/PSA PRODUCT_TYPE keyword
- Descriptions are extracted from PSA and PDS4 documentations, with comments.

### 2.1.3 Target description

**target_name** The target_name parameter identifies a target by name or ID. The target may be any Solar System body, exoplanet, planetary sample, or meteorite, plus in some cases astronomical objects or spacecraft. Any other feature (craters, regions, atmospheric layers...) must be named using the optional feature_name parameter (see 2.2.3). This parameter can be multivalued only to describe several targets related to a granule (e.g., with events). Alternative names of the same target must not be listed here, but may be provided through the optional alt_target_name parameter. In some rare cases no target name can be defined, so this parameter can remain empty.

The best practice is to use the official designation of the target as defined by IAU. This parameter is case sensitive (mixing lower/upper cases) and all values must use the standard
spelling and case; unusual characters (such as intermediate spaces) are allowed, except quotes and hashes (preferably changed to _). Data providers must be aware that services which do not expose the IAU designations will not return answers to queries using them. Conversely, users must be aware that some data of interest might not be visible if they do not use the recommended IAU nomenclature for planetary bodies. The quaero name resolver\(^2\) from IMCCE may help data providers (as well as users) to handle multiple denominations; it is available from the VESPA portal to support queries.

Concerning celestial objects (at fixed position, i.e., stars, galaxies...) the name should be identifiable through Simbad\(^3\).

Other best practices are listed below:

- The Exoplanet Encyclopedia provides a nearly complete list of currently known extrasolar planets: [http://exoplanet.eu/index.php](http://exoplanet.eu/index.php) (also available as an EPN-TAP service)
- The catalog of lunar samples is available here: [http://www.lpi.usra.edu/lunar/samples/](http://www.lpi.usra.edu/lunar/samples/)
- Other planetary samples are listed in topical web sites, e.g., samples from the Stardust mission are described here: [http://curator.jsc.nasa.gov/stardust/catalog/](http://curator.jsc.nasa.gov/stardust/catalog/)
- Consortia such as the IGSN\(^4\) issue IDs for samples
- Asteroids: Usage is to use preferably name (if it exists) or principal designation (the number is not used here, it can be included in alt_target_name)
- Meteors: the IAU Meteor Data Center assigns IDs to meteors and meteor showers
- Calibration targets: values can relate to existing names in a given archive (e.g., the PSA contains values such as bias, checkout, dark, flatfield, internal source...)

**Example TAP queries:**

```
SELECT TOP 1000 * FROM spectro_asteroids.epn_core
WHERE (target_name LIKE '%Ceres%' OR target_name LIKE '%Vesta%')
AND (ivo_hashlist_has(target_class, 'dwarf_planet') = 1 OR
    ivo_hashlist_has(target_class, 'asteroid') = 1)
```

---

\(^2\)https://ssp.imcce.fr/webservices/ssodnet/api/quaero/
\(^3\)http://simbad.u-strasbg.fr/simbad/
\(^4\)https://www.igsn.org
Will return data only from 1 Ceres or 4 Vesta.

The LIKE operator is used here to handle cases when 1) the name includes a number, either ‘1 Ceres’ or ‘(1) Ceres’, which (although not recommended) may occur in some services, 2) the special case of dwarf planets, which may be tagged as asteroids, or as ‘dwarf_planet#asteroid’ (recommended, see below).

The usage of LIKE cannot be generalized to all target_class values, because of ‘planet’ which also responds to ‘dwarf_planet’ and ‘exoplanet’. The ivo_hashlist_has function smoothly handles all values of target_class. Also beware that using LIKE on target_name may generate false positives.

Example:

1P is the official IAU designation for comet Halley

target_class  The target_class parameter identifies the type of the target. Solar System bodies are defined without ambiguity by the couple target_class and target_name. In other cases, targets may have no proper name (i.e., samples), but the target_class parameter must contain a value in all cases. The possible values for target_class are:

- asteroid
- dwarf_planet
- planet
- satellite
- comet
- exoplanet
- interplanetary_medium
- sample
- sky
- spacecraft
- spacejunk
- star
- calibration

Usage:
• Most Solar System bodies are expected to belong to a single target class, and the pair of values is required to resolve homographies (such as Io). However, classifications may evolve with time.

• It is therefore good practice to use a hash-separated-list to handle any ambiguity or evolution in target classes, such as dwarf planets (dwarf_planet#asteroid) or transitional objects (comet#asteroid). This will help support queries by target_class in particular.

• If several target_name values are provided for a granule, several target_class may be present as hash-separated values.

• “interplanetary_medium” refers in particular to interplanetary dust observed in context (not to samples), or to simulations (e.g., of the primordial nebula). Details can be provided through target_region.

• “sample” refers to lunar or planetary samples, to meteorites, but also to terrestrial samples, e.g., in laboratory studies.

• “satellite” stands for natural satellites only — artificial satellites are handled though spacecraft or spacejunk.

• “star” is used typically for calibration targets, and for the Sun.

• “sky” should be used for all other celestial bodies, usually referred to by their sky coordinates. It also includes the Interstellar Medium.

• “calibration” is used for observations only related to instrument or signal calibration, including dark current, flat field, reference sample (in lab), etc (use of “calibration” with planetary bodies is left to data providers).

• planetary rings are not considered a target class, but appear as target_region = “ring” with their planet indicated in target_name.

Comment:
All “Types” of objects used by the quaero resolver are included in the EPNCore list: “asteroid”, “spacecraft”, “comet”, “exoplanet”, “spacejunk”, “satellite”, “planet”, “dwarf planet”, “star”.

Additional EPNCore classes are: “interplanetary_medium”, “sample”, “sky”, “calibration”.

2.1.4 Axes

EPNCore describes the data coverage along 8 main axes: time, spectral, spatial (3 coordinates), and viewing geometry (3 angles).

Coverages along the time/space/spectral axes are described with a range and a resolution and/or sampling step.

Each quantity is a set of 2 parameters providing min and max values. They must both be present with the same value when min = max. Additional single parameters provide extra information: spatial_frame_type, s_region, and the optional coverage parameter.
time (min/max)  The time_min and time_max parameters provide the date and time of acquisition in the observer frame.

The time parameters are always provided in UTC and formatted in Julian Days (expressed in double precision). EPNCore uses standard JD to avoid ambiguity with time origin (as opposed to ObsCore which uses Modified JD). The use of double precision floats ensures an accuracy on the order of 1 ms, which is considered sufficient for search purposes (the initial accuracy is preserved in the data itself). A client front-end may expose times formatted in a more convenient way for human users.

Example TAP queries:

Search data contained in a time range:

```
SELECT TOP 100 * FROM spectro_asteroids.epn_core
WHERE time_min > 2455197.5 AND time_max < 2455927.5
```

Search data described by a start time parameter:

```
SELECT TOP 100 * FROM spectro_asteroids.epn_core
WHERE time_min BETWEEN 2455197.5 AND 2455927.5
```

Time range is, by default, provided in the observer frame (i.e., at the observer location at that moment), which is almost always the native time in the data. For instance, space-borne observations are usually documented with spacecraft on-board time, which is expected here (provided as JD, not as on-board clock timing). For other cases, the location where time is measured must be provided through the time_refposition parameter.

To support space-borne vs ground-based campaigns, multiple spacecraft observations, or survey of periodic events, measurement times need to be corrected for light path. Whenever comparisons are potentially involved, the use of the target_time_min and target_time_max parameters is recommended in addition to time_min and time_max to make this kind of comparison simpler.

Non-compulsory parameters may be used to accommodate additional, specially formatted time scales such as native on-board time for space borne observations. The information of the time_min and time_max parameters is however highly recommended for all observations, as it is used by default to search data collections.

time_sampling_step (min/max)  These parameters provide the sampling step in seconds for measurements of dynamical phenomena, and for computations. This is the time between 2 successive measurements or data, which is mostly relevant when the measurements are regularly spaced. This may also reflect an input parameter, e.g., for simulations or ephemeris computations. This parameter is intended to allow the user to search for time-resolved observations of dynamic phenomena, but can also accommodate the “repetition time” for several types of instruments.
**time_exp (min/max)**  These parameters correspond to the integration time (or exposure time) of measurements in seconds. They provide an estimate of the time resolution for dynamical phenomena, as well as an indication of relative S/N ratio inside a given data collection. This time is usually shorter than the time_sampling_step if both are present. It provides the overall integration time, i.e., individual exposure time × number of summed frames when relevant.

**spectral_range (min/max)**  The spectral_range parameters define the upper and lower bounds of the spectral domain of the data. This quantity is conventionally expressed on a frequency scale in Hertz, although the client front-end may propose conversions to other units/scales. The spectral range and associated parameters only apply to electromagnetic waves. See the optional parameters from the particle spectroscopy extension for particle energy or mass detection.

Since this is the standard parameter to identify a spectral range, it is recommended to fill it even for images obtained through a filter (for instance, by giving the central wavelength ± FWHM/2, or even just the central wavelength as both minimum and maximum, which may already be enough to make the granule findable). The SVO Filter Profile Service\(^5\) provides responses for many instrument filters; bandwidths can be retrieved easily with a python library: [https://github.com/hover2pi/svo_filters](https://github.com/hover2pi/svo_filters).

Care must be taken when computing spectral ranges from metadata given as wavelength. For instance, when the spectral range \(\lambda\) is given in micrometers, with the constants from sect. 3.2 one would compute (notice the change of roles!)

\[
\begin{align*}
\text{spectral_range}_\text{min} &= 2.99792458 \cdot 10^{14}/\lambda_{\text{max}} \\
\text{spectral_range}_\text{max} &= 2.99792458 \cdot 10^{14}/\lambda_{\text{min}}
\end{align*}
\]

**spectral_sampling_step (min/max)**  The spectral_sampling_step parameters provide the spectral separation between the centers of two adjacent filters or channels. Similar to the spectral_range quantities, they are expressed on a frequency scale in Hz.

These parameters are most relevant for radio observations (long wavelengths / low frequencies) with regular sampling. They can also help distinguish between Nyqvist and sub-Nyqvist sampling rates (together with resolution).

Again, care must be taken when converting sampling steps from non-frequency metadata. For example, when the spectral sampling step is given in micrometers of wavelength, \(\Delta \lambda\), with the constants from sect. 3.2 one would compute

\[
\text{spectral_sampling_step}_\ast = 2.99792458 \cdot 10^{14} \Delta \lambda_{\mu} / \lambda_{\mu}^2.
\]

For wavelengths in nanometers, \(\lambda_{\text{nm}}\), this becomes

\[
\text{spectral_sampling_step}_\ast = 2.99792458 \cdot 10^{11} \Delta \lambda_{\text{nm}} / \lambda_{\text{nm}}^2.
\]

Finally, for a wave number \(u\) in cm\(^{-1}\), the conversion is

\[
\text{spectral_sampling_step}_\ast = 2.99792458 \cdot 10^6 \Delta u
\]

\(^5\)http://svo2.cab.inta-csic.es/theory/fps3/
**spectral_resolution (min/max)** The spectral_resolution parameters provide the (dimensionless) resolving power $\lambda/\Delta\lambda = \nu/\Delta\nu$.

These parameters are mostly intended to provide an order of magnitude, e.g., to distinguish between Fourier spectrometers, grating spectrometers or filter cameras, or between observations related to surfaces or atmospheres. This is often most relevant for optical/infrared observations (short wavelengths / high frequencies).

Note that when computing the spectral resolution using wavelengths, minimum and maximum change roles, as in

$$
spectral\_resolution\_min = \frac{\lambda_{\text{min}}}{\lambda_{\text{max}}} (\Delta\lambda)$$
$$
spectral\_resolution\_max = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}} (\Delta\lambda)
$$

**c1 (min/max)**
**c2 (min/max)**
**c3 (min/max)** These parameters provide up to three spatial coordinates of the measured target (note that these parameter names contain no underscore before min/max, in contrast with the other ones). The coordinates depend on the spatial_frame_type parameter defined below. All services must handle three spatial coordinates, even if the third one is always set to NULL. Some coordinates are measured along a circle and must handle 0-meridian crossing, therefore min/max values are actually start/stop values defining the footprint. In body-fixed coordinates, c1 is longitude and is always counted eastward; c1min is therefore the westernmost longitude and c1max the easternmost one; the opposite applies to Right Ascension on the sky. Note that the c3 parameter is related to the observed area, relative to a reference surface in most cases; the target distance (e.g., geocentric distance for ground based observations, or spacecraft distance) is introduced by the optional parameters “earth_distance” or “target_distance”.

*Example TAP queries:

Requests on C1/C2 coordinates often involve comparisons between two bounding boxes. Below are typical examples in body-fixed coordinates, when poles are not included in the bounding box (note that this type of query is more compact and more accurate if parameters s_region or coverage are provided).

**Increasing longitudes: data range intersects [100, 200]:**

```sql
SELECT TOP 100 * FROM vvex.epn_core
WHERE ((c1min <= c1max AND c1min <= 200.0 AND c1max >= 100.0) OR
(c1min > c1max AND (c1min <= 200.0 OR c1max >= 100.0)))
AND c2max >= -20.0 AND c2min <= 20.0 AND spatial_frame_type = 'body'
```

**0-crossing longitude range: data range intersects [300, 20]:**

```sql
SELECT TOP 100 * FROM vvex.epn_core
WHERE (c1min <= 20.0 OR c1max >= 300.0 OR c1min > c1max)
AND c2max >= -20.0 AND c2min <= 20.0 AND spatial_frame_type = 'body'
```
Increasing longitudes: data range is included in [100, 200]:

```sql
SELECT TOP 100 * FROM vvex.epn_core
WHERE c1min >= 100.0 AND c1max <= 200.0 AND c1min <= c1max
AND c2max <= 20.0 AND c2min >= -20.0 AND spatial_frame_type = 'body'
```

0-crossing longitude range: data range is included in [300, 20]:

```sql
SELECT TOP 100 * FROM vvex.epn_core
WHERE ((c1min >= 300.0 AND c1max <= 20.0) OR
(c1min >= 300.0 AND c1max >= 300.0) OR
(c1min <= 20.0 AND c1max <= 20.0))
AND c2max <= 20.0 AND c2min >= -20.0 AND spatial_frame_type = 'body'
```

In order to make uniform requests possible, spatial coordinates provided in the EPNCore table must be standardized. However, they can be provided in several systems types, as defined by the spatial_frame_type parameter. The native coordinate system used in the EPNCore table can be described by parameters spatial_coordinate_description and spatial_origin. This is intended to provide this information prior to loading the data, especially when several coordinate systems are available in the same service. Descriptions for EPN-TAP are provided in a separate document (see extension and vocabulary page [https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions) under Planetary Coordinate Systems). Secondary coordinates can be introduced using additional parameters, e.g., c1 and c2 ranges identify the visible region of a planetary disk, while extra RA / Dec parameters provide location on the sky at this moment, and subobserver_longitude_min/max and subobserver_latitude_min/max provide the coordinates of the disk center.

- **c1_resol (min/max)**
- **c2_resol (min/max)**
- **c3_resol (min/max)** These parameters provide a simple estimate of the resolution in the same frame/units as c1/c2/c3. For instance, if spatial_frame_type = ‘celestial’, it can accommodate the FWHM of the PSF on the sky (in degrees). The client front-end may propose more appropriate units to the user, depending on context (e.g., angular resolution in mas, distance in m...).

- **spatial_frame_type** Provides the “flavor” of the coordinate system, which defines the nature of the spatial coordinates (c1,c2,c3) in the EPNCore table and queries, and the way they are defined. A value is always required (use “none” if not applicable). This may be different from the coordinate system associated to / included in the data themselves. The reference frame itself is defined by the spatial_coordinate_description parameter, and the frame center can be specified using the spatial_origin parameter in case of ambiguity. The possible types are described below:

  - **celestial**: 2D angles on the sky, i.e., right ascension c1 and declination c2 plus possibly the heliocentric distance in c3 (in AU); although this is a special case of a spherical
frame, the order and conventions are different. Right ascension is provided in degrees. ICRS coordinates are assumed by default, but other frames may exist, e.g., HPC (Helio-Projective Cartesian) is centered on the Sun; such frames are identified from the spatial_coordinate_description parameter. For ground-based observations, Earth distance may be provided in the earth_distance parameter (not in c3).

- **body**: 2D angles in body-fixed frame: longitude c1 and latitude c2 plus possibly altitude as c3.
  
  A planetocentric system with eastward longitudes in the range \((0,360)^\circ\) is required for service interoperability. IAU 2009 planetocentric convention applies, in particular eastward longitudes and a north pole located on the north side of the invariant plane of the Solar System for planets and satellites. Unless otherwise specified in spatial_coordinate_description, this will be interpreted as the current IAU frame at the moment of access, in particular for definition of the prime meridian (Archinal and Acton et al., 2018).
  
  Parameter c3 is measured above the reference ellipsoid, and can be <0 for interiors. Two other parameters are available to provide other vertical scales if needed (radial_distance and altitude_fromshape).
  
  Planetocentric rotating frames are defined as spherical (rather than body).

- **cartesian**: \((x,y,z)\) as \((c1,c2,c3)\). This provides spatial coordinates in km.

- **spherical**: \((r, \theta, \phi)\) as \((c1,c2,c3)\), as defined in ISO standard 80000-2:2009; \(r = \) radius; \(\theta = \) zenith angle/collatitude/inclination; \(\phi = \) azimuth (E longitude). Angles are provided in degrees. When related to the sky or tied to a solid body, “celestial” (with RA/Dec) or “body” (with E longitude/latitude) frames must be used instead.

- **cylindrical**: \((r, \theta, z)\) as \((c1,c2,c3)\); \(r = \) radius in km; \(\theta = \) azimuth; \(z = \) height in km. The angle is provided in degrees.

- **none**: to be used when no spatial frame is defined for the data. This value is intended to prevent useless searches when space coordinates are not defined.

This parameter, although related to the specific coordinate system in use, is mostly intended to identify the nature of the coordinates provided in the service table (e.g., angles versus distances). This parameter is provided as a column of the EPNCore table to ensure it can be queried through the basic TAP mechanism. Although it will in general remain constant along the table for simple services, this parameter can vary from granule to granule and a value must therefore be provided in any query that includes spatial coordinates. Whenever additional coordinates are provided, they must be stored in extra columns of the table. If several different frames are mixed to provide the main coordinates, the use of different granule_gid may help clarify the situation. At any rate, easy access to the data must be considered during the design of the service.

**incidence (min/max)** The incidence parameters define the upper and lower bounds of the incidence angle range in the data (also known as Solar Zenith Angle). This is always indicated in
decimal degrees, and usually ranges from 0 to 90° (with 0° indicating the normal to the surface). Incidence and emergence angles may be counted relative to the normal of the ellipsoid model, or to the local normal (e.g., using a 3D shape model). In case the two systems are included in the data, these parameters introduce the values relative to the ellipsoid (local values may be provided through non-compulsory parameters).

**emergence (min/max)** The emergence parameters define the upper and lower bounds of the emergence, emission, or viewing angle range in the data. This is always indicated in decimal degrees, and usually ranges from 0 to 90° (with 0° indicating the normal to the surface). Incidence and emergence angles may be counted relative to the normal of the ellipsoid model, or to the local normal (e.g., using a 3D shape model). In case the two systems are included in the data, these parameters introduce the values relative to the ellipsoid (local values may be provided through non-compulsory parameters).

**phase (min/max)** The phase parameters define the upper and lower bounds of the phase angle range in the data (i.e., scattering angle \(-180°\), or light source – target-observer angle. It is always indicated in decimal degrees, and may range from \(-180°\) to \(180°\) (with 0° corresponding to opposition, i.e., light source in the back of the observer). Negative values may refer, e.g., to geometry before opposition, depending on context. Phase \(\phi\), incidence \(i\) and emergence \(e\) are partly related:

\[ |i - e| < |\phi| < i + e \]

If the azimuth angle \(a\) is available instead of the phase angle, the latter can be derived from knowledge of the three angles:

\[ \cos(\phi) = \cos(i) \cos(e) + \cos(a) \sin(i) \sin(e) \]

**s_region** This parameter introduces a footprint as a contour. The s_region parameter should be given for geolocalized data products in 2D, most notably on the sky (using RA, Dec) or on planetary surfaces (using E longitude, latitude) to communicate the geometry of a feature or an observation footprint. This is a single parameter with no min/max declinations. Coordinates are provided in the same reference system as c1/c2 parameters, clients must therefore inspect spatial_frame_type before interpreting such contours. Any type of contour that works with ADQL’s geometry operators (CONTAINS, INTERSECTS...) is legal here.

The coverage parameter introduces another type of footprint, with potential extension to time coverage. When both are present, care must be taken that they are consistent.

In the VOTable serialization the s_region parameter is represented as either an STC-S string value with xtype='adql:REGION' or a DALI 1.1-compliant numeric array value with xtype='point', 'circle' or 'polygon'.

**Implementation notes:**

Two syntaxes are possible:
1) The original ObsCore s_region one (currently limited to polygons) — a string with syntax: 
‘{(lon1,lat1), (lon2,lat2), ... }’ (with no quotes) where (lon1,lat1) = (10.d, 5d), character d 
included (it stands for degrees). Pairs (lon1,lat1) must sample the contour in a sequence provided 
in direct (counter-clockwise) sense as viewed from the observer. Notice that the result of a TAP 
query is an STC-S string, which has different format (as in ObsCore).

2) Modern DALI 1.1 geometries — a numeric array providing either a point (‘lon lat’, with 
no quotes), a circle (‘lon lat radius’, with no quotes), or a polygon (sequence of at least 3 ‘lon 
lat’ pairs, with no quotes). For polygons, pairs (lon lat) must sample the contour in a sequence 
provided in direct (counter-clockwise) sense as viewed from the origin (when on the sky) or from 
the observer (when on planets).

2.1.5 Data origin

**instrument_host_name**  This parameter provides the name of the observatory, spacecraft, or 
facility that performed the measurements. A hash-separated-list of host names must be provided 
for integrated data sets. In the EPNCore table, the acronym is preferred to the full name to 
avoid long strings and related errors (however, both values can be provided in the list). An 
observatory list is being developed in Europlanet 2024/VESPA and by merging various sources. 
For lab measurements, this is typically the name of the facility.

An Observatory and Facility Database\(^6\) is under study at IVOA, and will eventually be used as 
a reference. Reference lists that can be used in the meantime include:

- for ground-based observations, the list of IAU observatory codes:  
  http://www.minorplanetcenter.net/iau/lists/ObsCodesF.html 
  However, this list is not intended to include all ground-based observatories, and a comple-
  ment still needs to be identified (including e.g., radio telescopes).

- a reasonably complete list of radio telescopes is available here:  

- the WiseRep list also provides a database of telescopes and instruments:  
  https://wiserep.weizmann.ac.il/aux/telescopes  
  https://wiserep.weizmann.ac.il/aux/instruments

- Concerning space-borne data, the most complete list of international planetary missions 
  and orbital observatories is found here (included in a complete list of space missions with 
  ID): http://nssdc.gsfc.nasa.gov/nmc/ 
  Planetary missions are also listed here: http://nssdc.gsfc.nasa.gov/planetary/chronology. 
  html.

- Alternatively, the PDS dictionary defines values for many mission names:  
  http://pds.nasa.gov/tools/dictionary.shtml

\(^6\)https://voparis-wiki.obspm.fr/display/VES/Observatory+Facility+Database
Other mission names are supported by the SPICE system, but only as ID codes:

**instrument_name** Identifies the instrument(s) that acquired the data. A hash-separated-list of instruments must be provided for integrated data products. Service providers are invited to include multiple values for instrument name, e.g., complete name and usual acronym. This will allow queries on either “Visible Infrared Thermal Imaging Spectrometer” or “VIRTIS” to produce the same reply. For ground-based observations and lab measurements, this is typically the name of the instrument in use (not the telescope or facility). Non-compulsory parameters are available to provide more information about sensors, instrument modes, etc.

*Example:*

Visible Infrared Thermal Imaging Spectrometer#VIRTIS

An Observatory and Facility Database is under study at IVOA, and will eventually be used as a reference.

The most complete ready-to-use list of international planetary missions and orbital observatories is found here: http://nssdc.gsfc.nasa.gov/nmc/

Instruments on board planetary missions in particular are listed here: http://nssdc.gsfc.nasa.gov/nmc/experimentSearch.do

### 2.1.6 Granule call-back info

These parameters provide ancillary information for routine processing after a query. These 4 parameters must always contain a value. Dates are provided in ISO 8601 format.

**service_title** The schema name of the service: it provides a unique acronym for the service/table, constant throughout the service. It is intended to identify the source of the data in later stages, e.g., when handling multiservice results. When designing the service, care must therefore be taken to use a title not already ascribed to another EPN-TAP service. Special/unusual characters must be avoided (including #, ?, etc). Other identifiers (ivoid...) may be used in a future version.

Existing EPNCore services can be listed from the registry:

```sql
select distinct(schema_name), table_utype
from rr.res_schema natural
join rr.res_table
where table_utype LIKE ’%epn%’
```

**creation_date** Provides the date when the granule was introduced in the service.

---

7https://voparis-wiki.obspm.fr/display/VES/Observatory+Facility+Database
**modification_date**  Provides the date when the granule was last updated. This is intended to speed up mirroring between sites and to flag recalibrations. When unknown or not relevant, the creation date is replicated here.

**release_date**  Provides the date when the granule becomes public. This is intended to support a proprietary period. When unknown or not relevant, the creation date is replicated here.

### 2.2 Optional parameters

Optional parameters are predefined parameters (with unit and UCD) which are available to provide further information if needed. This section describes optional parameters of general use, while section 2.3 describes parameters related to specific science fields.

#### 2.2.1 Data Access Reference

If the data are not included in the table (i.e., provided in separate files), these 3 parameters must be present and contain a value. If alternative formats are also provided, they must be described in other granules with the same obs_id and possibly different group ID, or via datalink. Whenever the data consists of a few scalar fields, these parameters are best replaced by parameters providing the data itself in the table. This makes them usable in query constraints (e.g., mass, in a table providing the masses of Solar System bodies).

**access_url**  The data of interest are often stored in a file, not in the table itself. In this very usual case, the access_url parameter provides a complete path to the data products on the network, so that they are accessible for download by plotting or processing tools. This parameter links to a file, a web service (e.g., PlanetServer_CRISM service) or a script (e.g., Titan_profiles service); in such cases, the link must include the adequate arguments so that data is downloaded. This parameter must link to the actual data, not to a file of metadata nor a document (the datalink_url parameter can be used for this purpose).

**access_format**  Access_format provides the format of the data file linked through the access_url parameter.

The data may be stored under native format, and no format conversion is required to set up an EPN-TAP service; this field can therefore include reference to unusual formats. However, VO-ready formats are required to take advantage of visualization and processing in VO tools. Consistently with ObsCore, possible values are MIME-types written in lower case; the most common ones are listed on the extension and vocabulary page ([https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions)) under Data Formats and MIME Types.

**access_estsize**  The access_estsize field provides an order of magnitude of the size (in kilobytes) of the file available via the corresponding URL. It is intended to provide an indication that can help to tune download functionalities in an application, depending on data volume and transfer bit rate.
2.2.2 Miscellaneous file metadata

**thumbnail_url**  The thumbnail_url parameter contains the URL of a reduced version of the data product used for quick-look purpose (e.g., a small jpeg image, typically 200x200 pixels). This may be handy in the case of big data files or unusual data formats, to facilitate data selection by the user. EPN-TAP clients such as the VESPA portal use this thumbnail for an on-line quick-look. It therefore contributes significantly to a service’s accessibility. Preferred formats include jpeg and png, which are handled easily in a web browser. Larger or more elaborate previews must be provided as independent granules and identified via a granule_gid different from the data.

**file_name**  The file_name parameter introduces the name of the data file, with extension but no path information. In many data services, the file name encodes the most relevant metadata and may be a very handy access mechanism for specialists. In services providing sets of files in a complicated directory tree (e.g., related to a spacecraft operation plan), the file_name parameter is a handy key to perform automatic operations such as mirroring, pipeline processing, etc. A web service is available to retrieve a file from the file_name parameter in any EPN-TAP service (File grabbing interface (fgi)\(^8\)). Its use is therefore always recommended when data are provided in separate files, i.e., not included in the table. All file_name values in the EPNCore table are case sensitive and must reflect the filename of the product provided through access_url; for instance, if access_url is a link to a script converting an ascii file to VOTable, file_name must refer to the outcome of this script, in this case the VOTable.

**access_md5**  This parameter introduces a MD5 Hash for the file when available, to be used as a checksum.

**accref**  Although not an EPNCore parameter, this name is reserved for internal use and must not be used explicitly.

**datalink_url**  This parameter is used to provide extra accesses through a datalink/SODA interface. This interface can provide access, for instance, to previews, documentation, related data products (such as progenitors, calibration files, etc.) and to cut-out or processing services on the data.

*Implementation note:*

Datalinks can be parameterized with input values retrieved from the current granule at the time of ingestion. When multiple EPN-TAP parameters are required to build a datalink, they can be first concatenated in an extra parameter in the table (often called \(\text{ds\_id}\)).

\(^8\)https://voparis-wiki.obspm.fr/pages/viewpage.action?pageId=17236008
2.2.3 Supplementary description

bib-reference The bib-reference parameter introduces an individual bibliographic reference at granule level. This provides the origin of the data, e.g., if the resource is a compilation of data from various origins. References are best provided as bibcode\(^9\), DOI, or arXiv identifier\(^10\), although other forms are acceptable.

publisher This parameter refers to the publisher of the data service, who is not necessarily at the origin of the data. Provided as a free format string.

processing-level-desc This parameter provides further details about processing-level if needed, e.g., a specific encoding related to a data collection, or more details about partial calibrations. Provided as a free format string.

internal-reference The internal-reference parameter can be used to identify granules (or sets of granules) intimately related to the current one, e.g., in a service containing both observations and results of analysis of observation sets, internal-reference can be used to provide the set of observations used to compute a result. This contains a hash-separated-list of granule_uid in the same service (which implies that those never contain the # character).

This is specifically intended to provide internal references in services which would otherwise need to be split in several tables, and it must only be used as a last resort (clever use of obs_id and granule_gid is usually more efficient).

external-link The external-link parameter can be used to provide extra information that does not fit easily in the table, and is intended for human reading only. This parameter must contain a single URL. This is typically an extended discussion of a granule on a web site, which may include images, tables, or other documents. For instance, the individual planet pages of the Encyclopedia of exoplanets are linked with this parameter, as they contain many links and sometimes web applets. Alternatively, the parameter can provide a link to a web service related to this granule; for instance, in the HRSC/MEx service (hrsc3nd) it displays a detailed view of the image with zooming functions for quick examination at full resolution.

species The species parameter introduces the chemical species of interest in simple data services. The formatting is very basic and simply uses the standard formula in ascii, e.g., H2O for water, CO2 for carbon dioxide or Fe for iron. This is one of the few query parameters provided in case sensitive form, using the standard chemical notation. This format can only accommodate atoms and simple molecular species, and does not support isotopic variations.

An example application is related to atmospheric composition: a table providing vertical profiles of gaseous species with altitude. All columns are abundances and are described by the same measurement-type parameter. The use of the “species” parameter allows identifying the

---

\(^9\)http://adsabs.github.io/help/actions/bibcode
\(^10\)https://arxiv.org/help/arxiv_identifier
various species and accessing the requested information. This of course is restrained to simple cases.

If more elaborated compositional information must be included, the use of parameter species_inchikey is recommended. InChiKeys identify molecule, including isotopes, but not the physical state or phase.

**messenger** This parameter is a generalization of waveband from VODataService (used e.g., in ConeSearch and SSA). It provides a rough indication of the spectral domain, e.g., when variable in a large archive. In addition, values for messenger can describe particles other than photons. Possible values are from the IVOA vocabulary (and case matters) [http://www.ivoa.net/rdf/messenger](http://www.ivoa.net/rdf/messenger).

**filter** This parameter introduces the standard name of a filter used during measurements. It is reserved for filters in imaging mode (no grating/grism #, etc). There is no predefined list, because of the large variety of possible denominations, but the best practice is to use a short and accurate ID. This VO service provides an extended list of references: [http://svo2.cab.inta-csic.es/svo/theory/fps3/](http://svo2.cab.inta-csic.es/svo/theory/fps3/)

This is intended to document the results of a search, rather than a search parameter. Therefore, it is recommended to also fill the spectral_range_min/max parameters to describe filter imaging — this is the only way to make filter imaging easily searchable.

**alt_target_name** This parameter introduces alternative names of the target, especially when they are more commonly used than the official IAU one (e.g., Halley vs 1P).

A frequent usage is to store a hash-separated-list of all alternative names for small bodies to avoid ambiguities (e.g., for asteroids: name, number, principal and provisional designations). Although this situation never occurs for small bodies, care must be taken to replace possible # characters by _ in target names (e.g., for samples).

**feature_name** Introduces a supplementary name to provide more details about the observed target. It is intended in particular to accommodate a local name (crater, surface feature, region...) whereas target_name is reserved to identify the whole body (Mars, Moon, Ceres...). Use of official features names defined by IAU ([http://planetarynames.wr.usgs.gov/](http://planetarynames.wr.usgs.gov/)) is preferred when relevant.

**target_region** Specifies a *type* of region on the target. This parameter only introduces generic regions (e.g., atmospheric layer, internal structure...), not specific local names which must be handled using the feature_name parameter.

Values are best selected from the IVOA rendition of the Unified Astronomy Thesaurus: [http://www.ivoa.net/rdf/urat](http://www.ivoa.net/rdf/urat)

Older sources (some of them may be integrated in the UAT) include:
Example:
“atmosphere”, “surface”, “ionosphere”, “ring”
The same values can be given in the subject keywords of the service’s registry record.

coverage This parameter introduces a footprint as a MOC (healpix-based). It is intended to support space/time coverages, most notably on the sky (assuming RA, Dec coordinates) or on planetary surfaces (assuming E longitude, latitude). This is a single parameter with no associated min/max parameters. It must contain a MOC 2.0 ascii string (MOC or STMOC). The result of the query must have xtype = “moc” (or possibly “stmoc” or equivalent in future standards) for proper handling in TAP.

The coverage parameter may appear together with s_region, which introduces a simpler type of footprint (2D contour). Coverage has precedence over s_region. When both are present, care must be taken that they are consistent.

2.2.4 Description of coordinate frame

spatial_coordinate_description
spatial_origin These two parameters provide a description of the spatial frame in use, depending on the spatial_frame_type parameter. Both parameters relate to the coordinates provided in the EPNCore table, not necessarily to those included in the data products.

Spatial_coordinate_description provides an acronym of the Coordinate Reference System as discussed in the extension and vocabulary page (https://hdl.handle.net/21.15110/epn_tap_extensions) under Planetary Coordinate Systems. For body-fixed frames, IAU/SPICE/WMS strings such as IAU2020:49900 are eligible — in this case, the final 00 which stands for planetocentric E-handed is the only valid option (refers to EPN-TAP standard on c1/c2 coordinates). If absent the current IAU coordinate system is assumed, depending on context.

Spatial_origin may be used to identify frame center in specific situations, using either target_name (referring to target center) or the IVOA vocabulary (http://www.ivoa.net/rdf/refposition). Other values may be used, e.g., to introduce specific geometries for simulations.

Examples:

spatial_coordinate_description: ICRS, IAU_MARS, IAU2000:49900
spatial_origin: Geocenter, Mars, (spacecraft)
Spatial coordinate description and an epoch derived from time_min/max can be used to feed the COOSYS element of VOTable 1.4 when its value is constant throughout the table (notice that most EPN-TAP coordinate frames are not standard values in this vocabulary at the time of writing):

```xml
<COOSYS ID="system" epoch="<epoch>" system="<spatial_coordinate_description/>
```

**time_refposition** Indicates where the time is measured, which can be a planet or a spacecraft — this does not handle time zones on Earth. This knowledge is required to cross-correlate event-based observations, in particular to indicate light-path differences. It applies to the time_min and time_max parameters (while target_time_min/max always refer to the target in the FoV). If this parameter provides no value, time is expected to be provided in the observer frame (related to instrument_host_name), which is not necessarily fixed in the Solar System.

Values for time_refposition may specify a target name (referring to target center) or spacecraft name, or use the IVOA vocabulary ([http://www.ivoa.net/rdf/refposition](http://www.ivoa.net/rdf/refposition)) including future versions of the hierarchy proposed at [https://wiki.ivoa.net/internal/IVOA/InterOpMay2019TDIG/topocenter.pdf](https://wiki.ivoa.net/internal/IVOA/InterOpMay2019TDIG/topocenter.pdf).

*Example:*

Geocenter, (solar system bodies), (spacecraft)

**time_scale** Provides the time scale applying to time_min and time_max parameters. UTC is assumed when no value is provided, since this is the expected value in most observational data services. Some services (in particular computational ones) may need to specify different time scales by using other using standard acronyms — values are preferably taken from the IVOA vocabulary: [http://www.ivoa.net/rdf/timescale](http://www.ivoa.net/rdf/timescale)

Both time_scale and time_refposition can be used to feed the TIMESYS element of VOTable 1.4 when their values are constant throughout the table (this is best avoided otherwise, e.g., when grouping coordinated observations from various locations in the Solar System):

```xml
<TIMESYS ID="time_frame" refposition="<time_refposition>"
    timeorigin="JD-origin" timescale="<time_scale/>
```

### 2.2.5 Target configuration and observing geometry

The next pairs of parameters provide additional location in time and space, and require min/max values. They are independent and can appear separately:

**solar_longitude (min/max)** Solar longitude (a.k.a. heliocentric longitude, or ecliptic longitude of the Sun, traditionally noted Ls) is the Sun-Planet vector angle counted from the planet position at northern hemisphere spring equinox. It provides a measurement of season.

Ls = 90° corresponds to the northern summer solstice, Ls = 180° to the northern autumn equinox, and Ls = -90° to the northern winter solstice. Although it is most usually applied to
Mars and Titan (using Saturn’s Ls), this notion can be enlarged to any planetary body without ambiguity.

This should not be confused with the true anomaly of the body (which is the same angle counted from the perihelion position), nor with the longitude of the subsolar point (see below).

**local_time (min/max)**  Provide the local time at the observed area. These parameters are provided in unit of target rotation divided by 24 and are measured from local midnight, i.e., in decimal (local) hours (range from 0 to 24, must increase with time at a given location).

**target_distance (min/max)**  The target_distance parameters introduce the distance of the observer to the observed area (in km) along the line of sight. This is mostly intended for space borne data, where it provides the spacecraft-target distance in km. For ground-based observations the earth_distance_min/max parameters should be used instead (in AU).

**target_time (min/max)**  The target_time parameters introduce the time measured in UTC scale at the target. This is intended to directly correlate simultaneous observations such as ground-based support and space-borne observations, or multi-spacecraft campaigns. Values are given as TIMESTAMP.

**earth_distance (min/max)**  **sun_distance (min/max)**  These two parameters provide the corresponding distance of Earth or Sun to the target at the time of observation (in AU). When the target is the Sun or Earth, use the target_distance_min/max parameters to provide the distance to the observer.

**subobserver_longitude (min/max)**  **subobserver_latitude (min/max)**  Provide coordinates of sub-observer point, in particular sub-Earth point (disk center) or central meridian for ground-based observations (this is different from the FoV location which is provided in c1/c2 parameters). The min/max pair is required to support long exposures related to time series, especially on giant planets to test target attitude.

**subsolarg Longitude (min/max)**  **subsolarg Latitude (min/max)**  Provide coordinates of sub-solar point, especially for disk images. The min/max pair is required to support long exposures related to time series.

**ra, dec**  Sky coordinates of the target can be provided in addition to standard coordinates, in which case they must be stored in these parameters. This may for instance document the location of a planet in a celestial image, while the main coordinates c1/c2 are used to provide the observed area as longitude and latitude. ICRS coordinate system is assumed; right ascension is provided in degrees (similar to ObsCore). RA and DEC parameters are interpreted correctly by most VO tools, therefore no min/max variations are allowed to maintain compatibility with existing software.
2.2.6 Vertical scales on planets

In body-fixed frame, parameters c3min/max, if present, introduce a vertical range counted from a reference surface, typically the reference ellipsoid for the current target.

Two other parameter pairs can be used provide different vertical scales, and must be considered for atmosphere-related services when available/relevant:

**radial_distance (min/max)**  Distance of observed area (at c1/c2) from body center, measured in km. Not to be confused with target_distance (which provides the distance from observer to observed area).

**altitude_fromshape (min/max)**  Altitude of observed area (at c1/c2) above local surface, measured in km. The local surface is provided by a DTM or a shape model. This parameter typically provides the height in the atmosphere.

Parameter c3 can be used to select services/data in a given altitude range above the ellipsoid, while radial_distance and altitude_fromshape provide other convenient vertical scales to compare observations from various origins. This use of c3min/max also applies to planetary interiors (c3 is then <0) and measurements at high altitude/distance.

2.3 Extensions

EPN-TAP extensions are living projects intended to support data services in new fields as they become available. As such, they are supported outside of this document and can be completed with additional parameters in the future, following agreement between data providers and the editors of this document.

Extensions are maintained on an external web page with a Permanent ID: [https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions) (PID provided by EUDAT/B2HANDLE).

The parameters listed in this section have however been agreed upon, and can be considered as part of the EPN-TAP standard. Parameters given here have fixed names, units, and UCDs. Extension pages cannot override them. Extension pages can, however, change the recommended descriptions (which are non-normative anyway), and they can deprecate parameters given here. Hence, data providers intending to use parameters discussed in this section should inspect the corresponding extension pages before adopting any of these parameters. Parameters from extensions can be used independently, like the generic optional parameters described in section 2.2.

2.3.1 Particle spectroscopy extension

These parameters are related to the spectral distribution of particles only – for electro-magnetic waves, the spectral_* parameters apply.

When used, these parameters define an extra axis and must all be present. This set defines an additional axis for particle energy, all with min/max values.
particle_spectral_type  The `particle_spectral_type` parameter specifies the type of axis in use: either ‘energy’ (provided in eV), ‘mass’ (in amu), or ‘mass/charge’ (in amu/qe).

particle_spectral_range (min/max)  The `particle_spectral_range` parameters define the upper and lower bounds of the spectral domain for particles. Depending on the `particle_spectral_type` parameter, this quantity is expressed on an energy, mass, or mass/charge scale, with respective units eV, amu, or amu/qe.

particle_spectral_sampling_step (min/max)  The `particle_spectral_sampling_step` parameters provide the spectral separation between measurements, in the same scale and unit as `particle_spectral_range`. This parameter is mostly intended to provide an order of magnitude.

particle_spectral_resolution (min/max)  The `particle_spectral_resolution` parameters correspond to the actual resolution of the measurements, and are provided in the same scale and unit as `particle_spectral_range`. This parameter is mostly intended to provide an order of magnitude.

2.3.2 Solar System Objects extension

Most services providing descriptions of solar system objects contain no observation files (only inferred properties). Dedicated parameters are defined for the most common quantities.

mean_radius, equatorial_radius, polar_radius  These parameters are used to provide solar system object sizes (in km)

diameter  Target diameter, or equivalent diameter for binary objects (in km)

mass  Solar system object mass (in kg)

sidereal_rotation_period  Solar system sidereal rotation period (in hour)

semi_major_axis, inclination, eccentricity, long_asc, arg_perihel, mean_anomaly  Provide the standard orbital parameters (distance in AU, angles in degrees)

epoch  Provides a date of interest in JD

magnitude, flux, albedo  Provide values of respective quantities (flux in mJy).

When `target_class` = asteroid, dwarf_planet or comet, the following parameters can be used:
**dynamical_class** introduces the class of small body, from an enumerated list. The draft list includes: TNO, MBA, NEO, OCC (Oort Cloud comet), JFC (Jupiter family comet), Centaur. See the extension and vocabulary page [https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions) under Small bodies sub-types.

**dynamical_type** introduces a subdivision of the above, from an enumerated list. It currently includes:

- NEO: Atira, Aten, Apollo, Amor (complete)
- TNO (complete, but values may be adjusted): res 2:5 , res 1:2, Plutino, Hot classical, Cold classical, Scattered disk object, Detached object, Inner Oort object

See the extension and vocabulary page [https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions) under Small bodies sub-types.

**taxonomy_code** provides taxonomy codes for small bodies, typically related to spectral properties.

### 2.3.3 Maps

**map_projection** Provides map projection description (preferably as FITS name or code) or parameters as a free string — for instance, several services use this to store proj4 parameters. This refers to the data, not necessarily to the spatial coordinates in the table (which have to be east-handed).

**map_height, map_width** These parameters give the number of pixels along the two axes of a raster map. This specification does not constrain the orientation of such maps, and hence 'width' and 'height' do not correspond to physical directions.

**pixelscale (min/max)** Pixel size at the target (usually at surface), in km/pixel.

**map_scale** A string providing scale as a ratio (e.g., "1:50000").

### 2.3.4 Contributive works

These parameters are intended to specify the data origin in services compiling amateur data, detection networks, experimental facilities, or any service distributing data provided by multiple sources.

**observer_name, observer_id, observer_code** Provide the name, ID and possibly internal code of the observer. Name is provided as free text.
observer_institute, observer_country  Provide the affiliation and country of residence of the observer, as free text. The later is mostly intended for amateur or pro-am contributions.

observer_location, observer_lon, observer_lat  Give broad location and geographic position of the observer or telescope. observer_location can be used when the exact location cannot be released (e.g., for small telescopes).

original_publisher  Services compiling data from many sources can use this parameter to refer to the source of the data.

Parameters from other extensions  Parameters from other extensions may be helpful in this context:

The data_calibration_desc parameter from the spectroscopy extension can be used to provide information on post-processing (this is preferred over a parameter named “comment”)

producer_name, producer_institute  provide reference to who produced the data, especially for experimental data services

2.3.5 Experimental spectroscopy

Specific *_desc parameters describe the sample and the experimental setup. Although optional, they are important to provide the context. They introduce free descriptive strings which are not intended as search parameters.

producer_name, producer_institute  Provide reference to who measured the sample, as free text.

sample_id  Additional identifier of the sample, e.g., a specific fraction of a meteorite (in addition to target_name). Intended to refer to a pre-existing catalogue of a collection, will therefore contain a name/ID mainly for local use.

sample_classification  Provides composition as group, class, sub-class, etc, of sample, concatenated in a hash-separated-list with no particular order. It should include the specification “meteorite” plus the meteorite type when applicable, as well as description of main mixtures ingredients. Meteorite types can be provided as in (Krot and Keil et al., 2005) or equivalent. Dana (Dana and Dana et al., 1997) or Struntz (Nickel, 2001) classification tags can be used for minerals. To reduce the likelihood of false positives, minor and trace components must not be included in sample_classification.

Examples below are taken from the PDS_speclib service:

Terrestrial mineral:

natural#solid#earth#mineral#unclassified#nesosilicate#unclassified#olivine

Lunar sample:

natural#solid#moon#rock#unclassified#igneous#unclassified#pyroxene
Martian meteorite:

- natural
- solid
- mars
- rock
- unclassified
- unclassified
- unclassified
- unclassified
- unclassified
- meteorite
- snc

**species_inchikey**  Provides an accurate machine-oriented description of species involved, as per IUPAC standard (Heller and McNaught et al., 2015). InChiKeys are fixed-length strings which do not include #, and are therefore consistent with the hash-separated-list syntax.

**grain_size (min/max)**  Provide the particle size range in $\mu$m. A very large value (eg, $>1000$ $\mu$m) can be used locally in a service to identify bulk material. This is really *grain_size*, since *particle* is reserved for elementary particle spectroscopy.

**azimuth (min/max)**  Provides the azimuth angle in degrees — see if negative values of angles can have a special meaning (also for phase angle)

**pressure, temperature**  Provide experimental conditions, in bar and K. Although it is not recommended by the IAU, the unit bar is accepted for pressure in this context, as it refers to terrestrial laboratory conditions.

**sample_desc**  Free string describing the sample, its origin, and possibly its preparation

**setup_desc**  Free string describing the experimental setup if needed — this may include the aperture (size of sample measured).

**data_calibration_desc**  Free string describing data post-processing / calibration.

**geometry_type**  Describes the geometry for spectral measurements, from an enumerated list. Possible values are maintained on the extension and vocabulary page ([https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions)) under Lab spectroscopy extension. Possible values currently include:

- **direct** (therefore in emission)
- **specular**
- **bidirectional**
- **directional-conical**
- **conical-directional**
- **biconical**
- **directional-hemispherical**
- **conical-hemispherical**
- hemispherical-directional
- hemispherical-conical
- bihemispherical
- directional
- conical
- hemispherical
- other geometry
- unknown

`spectrum_type` Explicitly provides the type of spectral measurement, from an enumerated list — this is where ‘radiance factor’, ‘reflectance’, ‘reflectance factor’, etc, are defined. The purpose is to give a precise description of complex measurements independently from UCDs, which are not expected to reach this level of description.

Possible values and corresponding UCDs are maintained on the extension and vocabulary page ([https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions)) under Lab spectroscopy extension. UCDs appear both under measurement_type and in the data files.

`measurement_atmosphere` Provides description of experimental conditions as a free string. Measurements under vacuum are indicated here with the word ‘vacuum’.

Use of general parameters When used with the lab spectroscopy extension, some general parameters need special interpretation:

- `thumbnail_url` Provides a link to a small spectral plot. Caution should be taken to maintain minimal readability in the small format used in the VESPA portal. Larger plots can be included as separate granules (previews).
- `datalink_url` External documents are often available to provide extra information in this context, e.g., chemical analysis or a descriptive image of the sample. Such documents can be considered an extension of the table for the current granule, as opposed to other data products which must be defined as different granules/rows. The best solution identified consists of providing such links via Datalink.
- `target_class` = ‘sample’, constant.
- `target_name` provides the name or ID of the sample. A value is mandatory in this case. It introduces the name of a meteorite or lunar sample when applicable. Various parts of the same sample can be indicated and described in sample_desc (such as “Location A”, etc) or in sample_id.
- `measurement_type` provides the type of measurement/scale as a UCD (e.g., REFF, I_over_F, etc...), and is completed by `spectrum_type` which is expected to be more accurate in general.
original_publisher (from the Contributive work extension) may be used to identify a source database in a collective service such as SHADE.

2.3.6 APIS extension

A specific extension has been designed for compatibility with the APIS service, so that all services in the field of planetary aurorae can be queried together and consistently. This extension contains a set of nearly 25 specific parameters providing ephemeris and configuration quantities, which are listed on the extension and vocabulary page (https://hdl.handle.net/21.15110/epn_tap_extensions) under APIS extension.

Some parameters defined for APIS are of more general interest and can be used by other services distributing observational data from a large facility:

- obs_mode, detector_name, opt_elem — relate to instrument characteristics
- north_pole_position, target_primary_hemisphere, target_secondary_hemisphere — describe target geometric configuration
- orientation, platesc — describe image orientation and scale on the sky
- measurement_unit — provides data unit

2.3.7 Events

VOEvents (Seaman and Williams et al., 2006) can be archived in EPN-TAP tables. The following extra parameters are available to complete their metadata:

**event_type** Provides a type of event from a pre-defined list. Possible values include:

- meteor_shower
- fireball
- lunar_flash
- comet_tail_crossing

A more complete list of values will be maintained separately.

**event_status** Following the VOEvent standard, this can be:

- prediction
- observation
- utility (e.g., local event related to an instrument, like a change of detector)
Following the VOevent standard, this can be:

- followup
- supersedes
- retraction (an alternative is to update the event lists)

**Use of general parameters**  
**access_url** is expected to link to a VOevent file with **dataprod-uct_type = ev**

  **access_format** = text/xml (for broad identification) or = application/x-voevent+xml

  **instrument_host_name, instrument_name** For VOEvents that are actually predictions, use instrument_host_name = “simulation” and instrument_name = reference of code used, including version number.

  **target_name** groups all targets in a hash-separated-list (e.g., Sun#Earth)

  **obs_id** is identical for related events

### 2.3.8 Other extensions

Other extensions will be defined by topical working groups. They will be maintained separately to complement the present document, and linked on the extension web page, with Permanent ID: [https://hdl.handle.net/21.15110/epn_tap_extensions](https://hdl.handle.net/21.15110/epn_tap_extensions) (PID provided by EU-DAT/B2HANDLE).

Extensions can be derived from independent Data Models, e.g., for exoplanets.
## 3 EPNCore Table

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Type</th>
<th>Unit</th>
<th>Description</th>
<th>UCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>granule_uid</td>
<td>Text</td>
<td></td>
<td>Unique ID in data service.</td>
<td>meta.id</td>
</tr>
<tr>
<td>granule_gid</td>
<td>Text</td>
<td></td>
<td>Common to granules of same type</td>
<td>meta.id</td>
</tr>
<tr>
<td>obs_id</td>
<td>Text</td>
<td></td>
<td>Associates granules derived from the same data</td>
<td>meta.id;obs</td>
</tr>
<tr>
<td>dataproduct_type</td>
<td>Text</td>
<td></td>
<td>Organization of the data product, from enumerated list</td>
<td>meta.code.class</td>
</tr>
<tr>
<td>measurement_type</td>
<td>Text</td>
<td></td>
<td>UCD(s) defining the data</td>
<td>meta.ucd</td>
</tr>
<tr>
<td>processing_level</td>
<td>Integer</td>
<td></td>
<td>Dataset-related encoding, or simplified CODMAC calibration level</td>
<td>meta.calibLevel</td>
</tr>
<tr>
<td>target_name</td>
<td>Text</td>
<td></td>
<td>Standard IAU name of target (must match target_class), case sensitive</td>
<td>meta.id;src</td>
</tr>
<tr>
<td>target_class</td>
<td>Text</td>
<td></td>
<td>Type of target, from enumerated list</td>
<td>src.class</td>
</tr>
<tr>
<td>time_min</td>
<td>Double</td>
<td>d</td>
<td>Start time (in JD). UTC measured at time_origin location (default is observer's frame)</td>
<td>time.start;obs</td>
</tr>
<tr>
<td>time_max</td>
<td>Double</td>
<td>d</td>
<td>Stop time (in JD). UTC measured at time_origin location (default is observer's frame)</td>
<td>time.end;obs</td>
</tr>
<tr>
<td>time_sampling_step_min</td>
<td>Float</td>
<td>s</td>
<td>Min time sampling step</td>
<td>time.resolution;stat.min</td>
</tr>
<tr>
<td>time_sampling_step_max</td>
<td>Float</td>
<td>s</td>
<td>Max time sampling step</td>
<td>time.resolution;stat.max</td>
</tr>
<tr>
<td>time_exp_min</td>
<td>Float</td>
<td>s</td>
<td>Min integration time</td>
<td>time.duration;obs.exposure;stat.min</td>
</tr>
<tr>
<td>time_exp_max</td>
<td>Float</td>
<td>s</td>
<td>Max integration time</td>
<td>time.duration;obs.exposure;stat.max</td>
</tr>
<tr>
<td>spectral_range_min</td>
<td>Float</td>
<td>Hz</td>
<td>Min spectral range (as frequency)</td>
<td>em.freq;stat.min</td>
</tr>
<tr>
<td>spectral_range_max</td>
<td>Float</td>
<td>Hz</td>
<td>Max spectral range (as frequency)</td>
<td>em.freq;stat.max</td>
</tr>
<tr>
<td>spectral_sampling_step_min</td>
<td>Float</td>
<td>Hz</td>
<td>Min spectral sampling step</td>
<td>em.freq;spect.binSize;stat.min</td>
</tr>
<tr>
<td>spectral_sampling_step_max</td>
<td>Float</td>
<td>Hz</td>
<td>Max spectral sampling step</td>
<td>em.freq;spect.binSize;stat.max</td>
</tr>
<tr>
<td>spectral_resolution_min</td>
<td>Float</td>
<td></td>
<td>Min spectral resolution (resolving power)</td>
<td>spect.resolution;stat.min</td>
</tr>
<tr>
<td>spectral_resolution_max</td>
<td>Float</td>
<td></td>
<td>Max spectral resolution (resolving power)</td>
<td>spect.resolution;stat.max</td>
</tr>
<tr>
<td>c1min</td>
<td>Float</td>
<td>Note 1</td>
<td>Min of first coordinate, depends on the frame</td>
<td>see table below</td>
</tr>
<tr>
<td>c1max</td>
<td>Float</td>
<td>Note 1</td>
<td>Max of first coordinate, depends on the frame</td>
<td></td>
</tr>
<tr>
<td>c2min</td>
<td>Float</td>
<td>Note 1</td>
<td>Min of second coordinate, depends on the frame</td>
<td></td>
</tr>
<tr>
<td>c2max</td>
<td>Float</td>
<td>Note 1</td>
<td>Max of second coordinate, depends on the frame</td>
<td></td>
</tr>
<tr>
<td>c3min</td>
<td>Float</td>
<td>Note 1</td>
<td>Min of third coordinate</td>
<td></td>
</tr>
<tr>
<td>c3max</td>
<td>Float</td>
<td>Note 1</td>
<td>Max of third coordinate</td>
<td></td>
</tr>
<tr>
<td>s_region</td>
<td>Text</td>
<td>Note 3</td>
<td>ObsCore-like footprint in 2D (if spatial_frame_type = celestial or body)</td>
<td>pos.outline;obs.field</td>
</tr>
<tr>
<td>c1_resol_min</td>
<td>Float</td>
<td>Note 2</td>
<td>Min resolution in first coordinate</td>
<td>Note 2</td>
</tr>
<tr>
<td>c1_resol_max</td>
<td>Float</td>
<td>Note 2</td>
<td>Max resolution in first coordinate</td>
<td>Note 2</td>
</tr>
<tr>
<td>c2_resol_min</td>
<td>Float</td>
<td>Note 2</td>
<td>Min resolution in second coordinate</td>
<td>Note 2</td>
</tr>
</tbody>
</table>

EPNCore mandatory parameters (Must be present, possibly empty) (bold face: a value is required)
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Unit</th>
<th>Description</th>
<th>UCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>c2_resol_max</td>
<td>Float</td>
<td>Note 2</td>
<td>Max resolution in second coordinate                                         Note 2</td>
<td></td>
</tr>
<tr>
<td>c3_resol_min</td>
<td>Float</td>
<td>Note 2</td>
<td>Min resolution in third coordinate                                          Note 2</td>
<td></td>
</tr>
<tr>
<td>c3_resol_max</td>
<td>Float</td>
<td>Note 2</td>
<td>Max resolution in third coordinate                                          Note 2</td>
<td></td>
</tr>
<tr>
<td>spatial_frame_type</td>
<td>Text</td>
<td>Note 1</td>
<td>Flavor of coordinate system, defines the nature of coordinates. From enumerated list. Use &quot;none&quot; if undefined. meta.code.class:pos.frame</td>
<td></td>
</tr>
<tr>
<td>incidence_min</td>
<td>Float</td>
<td>deg</td>
<td>Min incidence angle (solar zenithal angle)                                   pos.incidenceAng;stat.min</td>
<td></td>
</tr>
<tr>
<td>incidence_max</td>
<td>Float</td>
<td>deg</td>
<td>Max incidence angle (solar zenithal angle)                                   pos.incidenceAng;stat.max</td>
<td></td>
</tr>
<tr>
<td>emergence_min</td>
<td>Float</td>
<td>deg</td>
<td>Min emergence angle                                                          pos.emergenceAng;stat.min</td>
<td></td>
</tr>
<tr>
<td>emergence_max</td>
<td>Float</td>
<td>deg</td>
<td>Max emergence angle                                                           pos.emergenceAng;stat.max</td>
<td></td>
</tr>
<tr>
<td>phase_min</td>
<td>Float</td>
<td>deg</td>
<td>Min phase angle                                                              pos.phaseAng;stat.min</td>
<td></td>
</tr>
<tr>
<td>phase_max</td>
<td>Float</td>
<td>deg</td>
<td>Max phase angle                                                              pos.phaseAng;stat.max</td>
<td></td>
</tr>
<tr>
<td>instrument_host_name</td>
<td>Text</td>
<td></td>
<td>Standard name of the observatory or spacecraft                               meta.id;instr.obsty</td>
<td></td>
</tr>
<tr>
<td>instrument_name</td>
<td>Text</td>
<td></td>
<td>Standard name of instrument                                                  meta.id;instr</td>
<td></td>
</tr>
<tr>
<td>service_title</td>
<td>Text</td>
<td></td>
<td>Title of resource = schema name                                               meta.title</td>
<td></td>
</tr>
<tr>
<td>creation_date</td>
<td>Timestamp</td>
<td>Note 4</td>
<td>Date of first entry of this granule                                          time.creation</td>
<td></td>
</tr>
<tr>
<td>modification_date</td>
<td>Timestamp</td>
<td>Note 4</td>
<td>Date of last modification                                                    time.processing</td>
<td></td>
</tr>
<tr>
<td>release_date</td>
<td>Timestamp</td>
<td>Note 4</td>
<td>Start of public access period (set to creation_date if no proprietary period) time.release</td>
<td></td>
</tr>
</tbody>
</table>

**Common optional parameters**

- **access_url**: URL of the data file, case sensitive (additional files may be linked through datalink_url). Can point to a script. If present, next 2 parameters must also be present. meta.ref.url;meta.file
- **access_format**: RFC 2045 media type (mime), required to be all-lower case meta.code.mime
- **access_estsize**: Estimate file size in kbyte (with this spelling) phys.size;meta.file
- **access_md5**: MD5 Hash for the file when available (real file, not script) meta.checksum;meta.file
- **thumbnail_url**: URL of a thumbnail image with predefined size (png 200 pix, for use in a client only) meta.ref.url;meta.preview
- **file_name**: Name of the data file only, case sensitive meta.id;meta.file
- **datalink_url**: Provides links to files or services on the server meta.ref.url
- **bib_reference**: Bibcode or doi preferred; can be a URL or anything else. Refers to the granule meta.bib
- **publisher**: Resource publisher meta.curation
- **processing_level_desc**: Describes specificities of the processing level meta.note
- **internal_reference**: Related granule_uid(s) in the current service meta.id.cross
- **external_link**: Web page providing more details on the granule. meta.ref.url
- **species**: Identifies a chemical species, case sensitive meta.id;phys.atmol
- **messenger**: Electro-magnetic band, from enumerated list instr.bandpass
- **filter**: Identifies filter in use, typically for images meta.id;instr.filter
- **alt_target_name**: Provides alternative target name(s). Can be a hash list meta.id;src
- **target_region**: Type of region or feature of interest meta.id;src.obs.field
- **feature_name**: Secondary name (e.g., standard name of a region of interest) meta.id;src.obs.field
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Unit</th>
<th>Description</th>
<th>UCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td>Text</td>
<td></td>
<td>Introduces an ascii (ST)MOC (2D footprint, possibly including</td>
<td>pos.outline.obs.field</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>time, if spatial_frame_type = celestial or body)</td>
<td></td>
</tr>
<tr>
<td>spatial_coordinate_description</td>
<td>Text</td>
<td></td>
<td>ID of specific coordinate system and version / properties</td>
<td>meta.code.class.pos.frame</td>
</tr>
<tr>
<td>spatial_origin</td>
<td>Text</td>
<td></td>
<td>Defines the frame origin</td>
<td>meta.ref:pos.frame</td>
</tr>
<tr>
<td>time_reposition</td>
<td>Text</td>
<td></td>
<td>Defines where the time is measured (e.g., ground vs spacecraft).</td>
<td>meta.ref:time.scale</td>
</tr>
<tr>
<td>time_scale</td>
<td>Text</td>
<td></td>
<td>Always UTC in data services</td>
<td>time.scale</td>
</tr>
<tr>
<td>solar_longitude_min</td>
<td>Float</td>
<td>deg</td>
<td>Min Solar longitude Ls (location on orbit / season)</td>
<td>pos.ecliptic.lon:pos.helio-centric:stat.min</td>
</tr>
<tr>
<td>solar_longitude_max</td>
<td>Float</td>
<td>deg</td>
<td>Max Solar longitude Ls (location on orbit / season)</td>
<td>pos.ecliptic.lon:pos.helio-centric:stat.max</td>
</tr>
<tr>
<td>local_time_min</td>
<td>Float</td>
<td>h</td>
<td>Min local time at observed region</td>
<td>time.phase:time.period.rotation:stat.min</td>
</tr>
<tr>
<td>local_time_max</td>
<td>Float</td>
<td>h</td>
<td>Max local time at observed region</td>
<td>time.phase:time.period.rotation:stat.max</td>
</tr>
<tr>
<td>target_distance_min</td>
<td>Float</td>
<td>km</td>
<td>Min observer-target distance</td>
<td>pos.distance:stat.min</td>
</tr>
<tr>
<td>target_distance_max</td>
<td>Float</td>
<td>km</td>
<td>Max observer-target distance</td>
<td>pos.distance:stat.max</td>
</tr>
<tr>
<td>target_time_min</td>
<td>Timestamp</td>
<td></td>
<td>Min observing time in target frame</td>
<td>time.start:src</td>
</tr>
<tr>
<td>target_time_max</td>
<td>Timestamp</td>
<td></td>
<td>Max observing time in target frame</td>
<td>time.end:src</td>
</tr>
<tr>
<td>earth_distance_min</td>
<td>Float</td>
<td>AU</td>
<td>Min Earth-target distance</td>
<td>pos.distance:stat.min</td>
</tr>
<tr>
<td>earth_distance_max</td>
<td>Float</td>
<td>AU</td>
<td>Max Earth-target distance</td>
<td>pos.distance:stat.max</td>
</tr>
<tr>
<td>sun_distance_min</td>
<td>Float</td>
<td>AU</td>
<td>Min Sun-target distance</td>
<td>pos.distance:stat.min</td>
</tr>
<tr>
<td>sun_distance_max</td>
<td>Float</td>
<td>AU</td>
<td>Max Sun-target distance</td>
<td>pos.distance:stat.max</td>
</tr>
<tr>
<td>subobserver_longitude_min</td>
<td>Float</td>
<td>deg</td>
<td>Minimum sub-observer point longitude (sub-Earth for ground based observations)</td>
<td>pos.bodyrc.lon:stat.min</td>
</tr>
<tr>
<td>subobserver_longitude_max</td>
<td>Float</td>
<td>deg</td>
<td>Maximum sub-observer point longitude (sub-Earth for ground based observations)</td>
<td>pos.bodyrc.lon:stat.max</td>
</tr>
<tr>
<td>subobserver_latitude_min</td>
<td>Float</td>
<td>deg</td>
<td>Minimum sub-observer point latitude (sub-Earth for ground based observations)</td>
<td>pos.bodyrc.lat:stat.min</td>
</tr>
<tr>
<td>subobserver_latitude_max</td>
<td>Float</td>
<td>deg</td>
<td>Maximum sub-observer point latitude (sub-Earth for ground based observations)</td>
<td>pos.bodyrc.lat:stat.max</td>
</tr>
<tr>
<td>subsolar_longitude_min</td>
<td>Float</td>
<td>deg</td>
<td>Minimum sub-solar point longitude</td>
<td>pos.bodyrc.lon:stat.min</td>
</tr>
<tr>
<td>subsolar_longitude_max</td>
<td>Float</td>
<td>deg</td>
<td>Maximum sub-solar point longitude</td>
<td>pos.bodyrc.lon:stat.max</td>
</tr>
<tr>
<td>subsolar_latitude_min</td>
<td>Float</td>
<td>deg</td>
<td>Minimum sub-solar point latitude</td>
<td>pos.bodyrc.lat:stat.min</td>
</tr>
<tr>
<td>subsolar_latitude_max</td>
<td>Float</td>
<td>deg</td>
<td>Maximum sub-solar point latitude</td>
<td>pos.bodyrc.lat:stat.max</td>
</tr>
<tr>
<td>ra</td>
<td>Float</td>
<td>deg</td>
<td>Right ascension</td>
<td>pos.eq.ra:meta.main</td>
</tr>
<tr>
<td>dec</td>
<td>Float</td>
<td>deg</td>
<td>Declination</td>
<td>pos.eq.dec:meta.main</td>
</tr>
<tr>
<td>radial_distance_min</td>
<td>Float</td>
<td>km</td>
<td>Min distance from observed area to body center</td>
<td>pos.distance:pos.bodyrc:stat.min</td>
</tr>
<tr>
<td>radial_distance_max</td>
<td>Float</td>
<td>km</td>
<td>Max distance from observed area to body center</td>
<td>pos.distance:pos.bodyrc:stat.max</td>
</tr>
<tr>
<td>altitude_fromshape_min</td>
<td>Float</td>
<td>km</td>
<td>Min altitude of observed area above shape model / DTM</td>
<td>pos.bodyrc.alt:stat.min</td>
</tr>
<tr>
<td>altitude_fromshape_max</td>
<td>Float</td>
<td>km</td>
<td>Max altitude of observed area above shape model / DTM</td>
<td>pos.bodyrc.alt:stat.max</td>
</tr>
</tbody>
</table>

Parameters from extensions
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Unit</th>
<th>Description</th>
<th>UCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>obs_mode</td>
<td>Text</td>
<td></td>
<td>Observing mode</td>
<td>meta.code;instr.setup</td>
</tr>
<tr>
<td>detector_name</td>
<td>Text</td>
<td></td>
<td>Detector name</td>
<td>meta.id;instr.det</td>
</tr>
<tr>
<td>opt_elem</td>
<td>Text</td>
<td></td>
<td>Optical element name</td>
<td>meta.id;instr.det</td>
</tr>
<tr>
<td>instrument_type</td>
<td>Text</td>
<td></td>
<td>type of instrument</td>
<td>meta.id;instr.param</td>
</tr>
<tr>
<td>acquisition_id</td>
<td>Text</td>
<td></td>
<td>ID of the data file/acquisition in the original archive</td>
<td>meta.id;obs.proposal</td>
</tr>
<tr>
<td>proposal_id</td>
<td>Text</td>
<td></td>
<td>Proposal identifier</td>
<td>meta.id;obs.proposal</td>
</tr>
<tr>
<td>proposal_pi</td>
<td>Text</td>
<td></td>
<td>Proposal principal investigator</td>
<td>meta.id;PI;obs.proposal</td>
</tr>
<tr>
<td>proposal_title</td>
<td>Text</td>
<td></td>
<td>Proposal title</td>
<td>meta.title;obs.proposal</td>
</tr>
<tr>
<td>campaign</td>
<td>Text</td>
<td></td>
<td>Name of the observational campaign</td>
<td>meta.id;obs.proposal</td>
</tr>
<tr>
<td>target_description</td>
<td>Text</td>
<td></td>
<td>List of keywords</td>
<td>meta.note;src</td>
</tr>
<tr>
<td>proposal_target_name</td>
<td>Text</td>
<td></td>
<td>target name as in proposal title</td>
<td>meta.note;obs.proposal</td>
</tr>
<tr>
<td>target_apparent_radius</td>
<td>Float</td>
<td>arcsec</td>
<td>Apparent radius of the target</td>
<td>phys.angSize;src</td>
</tr>
<tr>
<td>north_pole_position</td>
<td>Float</td>
<td>deg</td>
<td>North pole (of target) position angle with respect to celestial north pole</td>
<td>pos.posAng</td>
</tr>
<tr>
<td>target_primary_hemisphere</td>
<td>Text</td>
<td></td>
<td>Primary observed hemisphere</td>
<td>meta.id;obs.field</td>
</tr>
<tr>
<td>target_secondary_hemisphere</td>
<td>Text</td>
<td></td>
<td>Secondary observed hemisphere</td>
<td>meta.id;obs.field</td>
</tr>
<tr>
<td>platesc</td>
<td>Float</td>
<td>arcsec/pix</td>
<td>pixel angular size or platescale (on sky only)</td>
<td>instr.scale</td>
</tr>
<tr>
<td>orientation</td>
<td>Float</td>
<td>deg</td>
<td>Position angle of image y axis (on sky only)</td>
<td>pos.posAng</td>
</tr>
<tr>
<td>measurement_unit</td>
<td>Text</td>
<td></td>
<td>Physical unit, same as Bunit in fits</td>
<td>meta.unit</td>
</tr>
<tr>
<td>observer_name</td>
<td>Text</td>
<td></td>
<td>Observer name</td>
<td>meta.id.PI;obs.observer</td>
</tr>
<tr>
<td>observer_id</td>
<td>Integer</td>
<td></td>
<td>Observer’s numeric identifier</td>
<td>meta.id.PI</td>
</tr>
<tr>
<td>observer_code</td>
<td>Text</td>
<td></td>
<td>Observer’s service username</td>
<td>meta.id.PI</td>
</tr>
<tr>
<td>observer_institute</td>
<td>Text</td>
<td></td>
<td>Observer institute</td>
<td>meta.note;obs.observer</td>
</tr>
<tr>
<td>observer_country</td>
<td>Text</td>
<td></td>
<td>Observer’s country of residence</td>
<td>meta.note;obs.observer</td>
</tr>
<tr>
<td>observer_location</td>
<td>Text</td>
<td></td>
<td>Broad location of the observer or telescope</td>
<td>pos;obs.observer</td>
</tr>
<tr>
<td>observer_lon</td>
<td>Float</td>
<td>deg</td>
<td>Observer’s approximate longitude</td>
<td>obs.observer;pos.earth.lon</td>
</tr>
<tr>
<td>observer_lat</td>
<td>Float</td>
<td>deg</td>
<td>Observer’s approximate latitude</td>
<td>obs.observer;pos.earth.lat</td>
</tr>
<tr>
<td>original_publisher</td>
<td>Text</td>
<td></td>
<td>Refers to the source of the data, e.g., in compilations of experimental data</td>
<td>meta.note</td>
</tr>
<tr>
<td>mean_radius</td>
<td>Float</td>
<td>km</td>
<td></td>
<td>phys.size.radius</td>
</tr>
<tr>
<td>equatorial_radius</td>
<td>Float</td>
<td>km</td>
<td></td>
<td>phys.size.radius</td>
</tr>
<tr>
<td>polar_radius</td>
<td>Float</td>
<td>km</td>
<td></td>
<td>phys.size.radius</td>
</tr>
<tr>
<td>diameter</td>
<td>Float</td>
<td>km</td>
<td>Target diameter, or equivalent diameter for binary objects</td>
<td>phys.size.diameter</td>
</tr>
<tr>
<td>mass</td>
<td>Float</td>
<td>kg</td>
<td>Mass of object</td>
<td>phys.mass</td>
</tr>
<tr>
<td>sidereal_rotation_period</td>
<td>Float</td>
<td>h</td>
<td>Object rotation rate</td>
<td>time.period.rotation</td>
</tr>
<tr>
<td>semi_major_axis</td>
<td>Float</td>
<td>AU</td>
<td></td>
<td>src.size.smajAxis</td>
</tr>
<tr>
<td>inclination</td>
<td>Float</td>
<td>deg</td>
<td>Orbit inclination</td>
<td>src.orbital.inclination</td>
</tr>
<tr>
<td>eccentricity</td>
<td>Float</td>
<td></td>
<td>Orbit eccentricity</td>
<td>src.orbital.eccentricity</td>
</tr>
<tr>
<td>long_asc</td>
<td>Float</td>
<td>deg</td>
<td>Longitude of ascending node, J2000.0</td>
<td>src.orbital.node</td>
</tr>
<tr>
<td>arg_perihel</td>
<td>Float</td>
<td>deg</td>
<td>Argument of Perihelion, J2000.0</td>
<td>src.orbital.periastron</td>
</tr>
<tr>
<td>mean_anomaly</td>
<td>Float</td>
<td>deg</td>
<td>Mean anomaly at the epoch</td>
<td>src.orbital.meanAnomaly</td>
</tr>
<tr>
<td>epoch</td>
<td>Double</td>
<td>d</td>
<td>Epoch of interest in JD</td>
<td>time.epoch</td>
</tr>
<tr>
<td>magnitude</td>
<td>Float</td>
<td>mag</td>
<td>Absolute magnitude. For small bodies, from HG magnitude system</td>
<td>phys.mag.Abs</td>
</tr>
<tr>
<td>flux</td>
<td>Float</td>
<td>mJy</td>
<td>Target flux</td>
<td>phot.flux.density</td>
</tr>
<tr>
<td>albedo</td>
<td>Float</td>
<td></td>
<td>Target albedo</td>
<td>phys.albedo</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Unit</td>
<td>Description</td>
<td>UCD</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>------------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>dynamical_class</td>
<td>Text</td>
<td></td>
<td>Class of small body, from enumerated list</td>
<td>meta.code.class;src</td>
</tr>
<tr>
<td>dynamical_type</td>
<td>Text</td>
<td></td>
<td>Subdivision of the class, from enumerated list</td>
<td>meta.code.class;src</td>
</tr>
<tr>
<td>taxonomy_code</td>
<td>Text</td>
<td></td>
<td>Code for target taxonomy</td>
<td>src.class.color</td>
</tr>
<tr>
<td>map_projection</td>
<td>Text</td>
<td></td>
<td>ID from enumerated list, or string with parameters (referring to a standard)</td>
<td>pos.projection</td>
</tr>
<tr>
<td>map_height</td>
<td>Float</td>
<td>pix</td>
<td>Map size in px</td>
<td>phys.size</td>
</tr>
<tr>
<td>map_width</td>
<td>Float</td>
<td>pix</td>
<td>Map size in px</td>
<td>phys.size</td>
</tr>
<tr>
<td>map_scale</td>
<td>Text</td>
<td></td>
<td>Preferably as a ratio (e.g., &quot;1:50000&quot;)</td>
<td>pos.wcs.scale</td>
</tr>
<tr>
<td>pixelscale_min</td>
<td>Float</td>
<td>km/pix</td>
<td>Min pixel size on a surface</td>
<td>instr.scale;stat.min</td>
</tr>
<tr>
<td>pixelscale_max</td>
<td>Float</td>
<td>km/pix</td>
<td>Max pixel size on a surface</td>
<td>instr.scale;stat.max</td>
</tr>
<tr>
<td>particle_spectral_type</td>
<td>Text</td>
<td></td>
<td>From enumerated list</td>
<td>meta.id;phys.particle;</td>
</tr>
<tr>
<td>particle_spectral_range_min</td>
<td>Float</td>
<td></td>
<td></td>
<td>phys.energy;phys.particle;</td>
</tr>
<tr>
<td>particle_spectral_range_max</td>
<td>Float</td>
<td></td>
<td></td>
<td>stat.min</td>
</tr>
<tr>
<td>particle_spectral_sampling_step_min</td>
<td>Float</td>
<td></td>
<td></td>
<td>spect.resolution;phys.particle;stat.min</td>
</tr>
<tr>
<td>particle_spectral_sampling_step_max</td>
<td>Float</td>
<td></td>
<td></td>
<td>spect.resolution;phys.particle;stat.max</td>
</tr>
<tr>
<td>particle_spectral_resolution_min</td>
<td>Float</td>
<td></td>
<td></td>
<td>spect.resolution;phys.particle;stat.min</td>
</tr>
<tr>
<td>particle_spectral_resolution_max</td>
<td>Float</td>
<td></td>
<td></td>
<td>spect.resolution;phys.particle;stat.max</td>
</tr>
<tr>
<td>producer_name</td>
<td>Text</td>
<td></td>
<td>Data producer name, especially in compilations of experimental data</td>
<td>meta.note</td>
</tr>
<tr>
<td>producer_institute</td>
<td>Text</td>
<td></td>
<td>Data producer institute, e.g., in compilations of experimental data</td>
<td>meta.note</td>
</tr>
<tr>
<td>sample_id</td>
<td>Text</td>
<td></td>
<td>Provides a local ID in an existing catalogue</td>
<td>meta.id;src</td>
</tr>
<tr>
<td>sample_classification</td>
<td>Text</td>
<td></td>
<td>Information related to class, sub-class, species... as hash list</td>
<td>meta.note;phys.composition</td>
</tr>
<tr>
<td>sample_desc</td>
<td>Text</td>
<td></td>
<td>Describes the sample, its origin, and possible preparation. Can be a hash list</td>
<td>meta.note</td>
</tr>
<tr>
<td>species_inchikey</td>
<td>Text</td>
<td></td>
<td>Fixed length string identifying the species. Can be a hash list</td>
<td>meta.id;phys.atmol</td>
</tr>
<tr>
<td>grain_size_min</td>
<td>Float</td>
<td>um</td>
<td>Min sample particle size</td>
<td>phys.size;stat.min</td>
</tr>
<tr>
<td>grain_size_max</td>
<td>Float</td>
<td>um</td>
<td>Max sample particle size</td>
<td>phys.size;stat.max</td>
</tr>
<tr>
<td>azimuth_min</td>
<td>Float</td>
<td>deg</td>
<td>Min azimuth angle for illumination</td>
<td>pos.azimuth;stat.min</td>
</tr>
<tr>
<td>azimuth_max</td>
<td>Float</td>
<td>deg</td>
<td>Max azimuth angle for illumination</td>
<td>pos.azimuth;stat.max</td>
</tr>
<tr>
<td>pressure</td>
<td>Float</td>
<td>bar</td>
<td>Ambient pressure</td>
<td>phys.pressure</td>
</tr>
<tr>
<td>measurement_atmosphere</td>
<td>Text</td>
<td></td>
<td>Describes experimental conditions. “vacuum” for measurements under vacuum.</td>
<td>meta.note;phys.pressure</td>
</tr>
<tr>
<td>temperature</td>
<td>Float</td>
<td>K</td>
<td>Ambient temperature</td>
<td>phys.temperature</td>
</tr>
<tr>
<td>setup_desc</td>
<td>Text</td>
<td></td>
<td>Describes the experimental setup. Can be a hash list</td>
<td>meta.note</td>
</tr>
<tr>
<td>data_calibration_desc</td>
<td>Text</td>
<td></td>
<td>Provides information on post-processing. Can be a hash list</td>
<td>meta.note</td>
</tr>
<tr>
<td>geometry_type</td>
<td>Text</td>
<td></td>
<td>Type of observation, from enumerated list. Can be a hash list</td>
<td>meta.note;instr.setup</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Unit</td>
<td>Description</td>
<td>UCD</td>
</tr>
<tr>
<td>--------------</td>
<td>--------</td>
<td>------</td>
<td>----------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>spectrum_type</td>
<td>Text</td>
<td></td>
<td>Type of spectral observation, from enumerated list TBD. Can be a hash list</td>
<td>meta.note;instr.setup</td>
</tr>
<tr>
<td>event_type</td>
<td>Text</td>
<td></td>
<td>Type of event from enumerated list</td>
<td>meta.code.class</td>
</tr>
<tr>
<td>event_status</td>
<td>Text</td>
<td></td>
<td>From enumerated list</td>
<td>meta.code.status</td>
</tr>
<tr>
<td>event_cite</td>
<td>Text</td>
<td></td>
<td>From enumerated list</td>
<td>meta.code.status</td>
</tr>
</tbody>
</table>
3.1 Table Notes

The entries in the Type column which are not otherwise annotated have the following meanings:

- **Text**: a string value represented in VOTable with a **datatype** of char and a suitable **arraysize**
- **Float**: a floating point value represented in VOTable with a **datatype** of float or double
- **Integer**: an integer value represented in VOTable with a **datatype** of int or long
- **Double**: a double precision floating point value represented in VOTable with a **datatype** of double

The following notes are referenced by number in the table:

1. Depending on context (as given by spatial_frame_type), see table 2. Longitude and RA range from 0 \(\cdots\) 360; Latitude and Dec range from \(-90\cdots+90\). When spatial_frame_type = "none", UCDs here are empty, and no unit is provided.

2. Spatial resolution parameters have the same unit as spatial coordinate parameters. The associated UCDs combine either pos.resolution (if linear) or pos.angResolution (if angular) with secondary stat.min or stat.max. 

3. Any contour type that works with ADQL’s geometry operators (CONTAINS, INTERSECTS...) is legal here.

4. Timestamps are provided as ISO-8601 strings as specified by DALI (Plante and Demleitner et al., 2017). On VOTable output, xtype="timestamp" is required.

3.2 Unit Conversions

In order to avoid significant changes when round-tripping between units customary in the various sub-disciplines and the ones given in the EPN-Core table (e.g., going from wavelength to frequency during ingestion and then back to wavelength for display purposes), services and clients MUST use the following constants when converting values.

The concrete values given here may change with minor versions of this standard. Clients should hence be prepared to take the protocol version (discoverable through the table utype) into account when doing unit conversion as new versions of EPN-TAP are issued.

- Speed of light in vacuum \(c = 299792458 \text{ m/s}\)
- Planck constant \(h = 6.62607015 \cdot 10^{-34} \text{ Js}\)
- The electronvolt \(1 \text{ eV} = 1.602176634 \cdot 10^{-19} \text{ J}\) (and the elementary electric charge this implies)
<table>
<thead>
<tr>
<th></th>
<th>celestial</th>
<th>body</th>
<th>cartesian</th>
<th>spherical</th>
<th>cylindrical</th>
</tr>
</thead>
<tbody>
<tr>
<td>c1min</td>
<td>pos.eq.ra; stat.min</td>
<td>pos.bodyrc.lon;stat.min</td>
<td>pos.cartesian.x;stat.min (in km)</td>
<td>pos.spherical.r;stat.min (in m)</td>
<td>pos.cylindrical.r;stat.min (in km)</td>
</tr>
<tr>
<td>c1max</td>
<td>pos.eq.ra; stat.max</td>
<td>pos.bodyrc.lon;stat.max</td>
<td>pos.cartesian.x;stat.max (in km)</td>
<td>pos.spherical.r;stat.max (in m)</td>
<td>pos.cylindrical.r;stat.max (in km)</td>
</tr>
<tr>
<td>c2min</td>
<td>pos.eq.dec; stat.min</td>
<td>pos.bodyrc.lat;stat.min</td>
<td>pos.cartesian.y;stat.min (in km)</td>
<td>pos.spherical.colat;stat.min</td>
<td>pos.cylindrical.azi;stat.min</td>
</tr>
<tr>
<td>c2max</td>
<td>pos.eq.dec; stat.max</td>
<td>pos.bodyrc.lat;stat.max</td>
<td>pos.cartesian.y;stat.max (in km)</td>
<td>pos.spherical.colat;stat.max</td>
<td>pos.cylindrical.azi;stat.max</td>
</tr>
<tr>
<td>c3min</td>
<td>pos.distance;stat.min (in AU)</td>
<td>pos.bodyrc.alt;stat.min (from surface only, implicitly from reference level, in km)</td>
<td>pos.cartesian.z;stat.min (in km)</td>
<td>pos.spherical.azi;stat.min</td>
<td>pos.cylindrical.azi;stat.max (height, in km)</td>
</tr>
<tr>
<td>c3max</td>
<td>pos.distance;stat.max (in AU)</td>
<td>pos.bodyrc.alt;stat.max (from surface only, implicitly from reference level, in km)</td>
<td>pos.cartesian.z;stat.max (in km)</td>
<td>pos.spherical.azi;stat.max</td>
<td>pos.cylindrical.azi;stat.max (height, in km)</td>
</tr>
</tbody>
</table>

Table 2: UCDs of the spatial columns depending on the value of spatial_frame_type.
4 EPN-TAP and the VO Registry

4.1 Registering EPN Data Collections and Services

EPNCore data collections are registered using VODataService (Demleitner and Plante et al., 2021) tablesets, where the table utype is set to

\[ \text{ivo://ivoa.net/std/epntap#table-2.0}. \]

This registration of the EPNCore table will work even if the table is exposed through protocols different from TAP. Normally, however, the tableset will be contained in a VODataService CatalogService record with a TAP capability, and this capability will be an auxiliary capability as per DDC (Demleitner and Taylor, 2019). For one-table services, a full TAPRegExt (Demleitner and Dowler et al., 2012) capability is also allowed.

Further capabilities, for instance for associated Datalink services, may be given in the same record.

An example for a registration record in VOResource, for the case of using an auxiliary capability referencing a main TAP service comes with this document\(^\text{11}\).

The noteworthy points in the record are:

- **A relationship** element referencing the main TAP service through which the service is queriable as per DDC:

\[
<\text{relationship}>
  <\text{relationshipType}>\text{served-by}</\text{relationshipType}>
  <\text{relatedResource} ivo-id="\text{ivo://org.gavo.dc/tap}"
    >\text{GAVO Data Center TAP service}</\text{relatedResource}>
</\text{relationship}>
\]

- The declaration for the auxiliary capability, including the access URL so clients do not need to follow the relationship just declared in simple applications:

\[
<\text{capability} standardID="\text{ivo://ivoa.net/std/TAP#aux}"/>
  <\text{interface} role="\text{std} version="1.1" xsi:type="\text{vs:ParamHTTP}"
    >\text{http://dc.zah.uni-heidelberg.de/tap}</\text{accessURL}>
  <\text{mirrorURL}>\text{https://dc.zah.uni-heidelberg.de/tap}</\text{mirrorURL}>
</\text{capability}>
\]

\(^{11}\text{https://www.ivoa.net/documents/EPNTAP/20220822/example-record.xml}\)
• Most importantly, the declaration of the table utype that lets clients discover that this particular table contains EPNCORE data:

```xml
<table>
  <name>mpc.epn_core</name>
  <title>EPN-TAP table for MPC Asteroid Orbital Data</title>
  <description>The EPN-TAP 2.0 version of...</description>
  <utype>ivo://ivoa.net/std/epntap#table-2.0</utype>
</table>
```

That in the example record, the resource description is identical to the description of the schema, which again is identical to the description of the table is an artefact of EPN-TAP registrations being single-table and is thus to be expected in most registrations of this type.

### 4.2 Discovering EPN-TAP services

Consumers in general are interested in TAP endpoints and table names for EPN-TAP services. By our registration pattern, this translates into resources with TAP capabilities that have a standard key for version 2 EPN-TAP in a table utype. During a transition period, clients are advised to also look for tables with the utype `ivo://vopdc.obspm/std/epncore#schema-2.0`.

Translated into RegTAP (Demleitner and Harrison et al., 2019), the following query would return TAP access URLs and the table names, using ADQL 2.1 CTEs for readability:

```sql
WITH epntables AS (
    SELECT ivoid, table_name
    FROM rr.res_table
    WHERE table_utype = 'ivo://vopdc.obspm/std/epncore#schema-2.0'
    OR table_utype LIKE 'ivo://ivoa.net/std/epntap#table-2.%%')

SELECT DISTINCT table_name, access_url
FROM epntables
  NATURAL JOIN rr.capability
  NATURAL JOIN rr.interface
WHERE standard_id LIKE 'ivo://ivoa.net/std/tap%%'
  AND intf_role = 'std'
```

The `DISTINCT` in the main query is a rough filter that removes entries duplicated because their tables are registered both in the main TAP record and in an auxiliary capability.

### A Changes from Previous Versions

This is the first public release.
EPN-TAP v1 was developed as an assessment study during the Europlanet-RI programme (2011), and was limited to test purposes.

References

http://doi.org/10.1007/s10569-017-9805-5

Bradner, S. (1997), ‘Key words for use in RFCs to indicate requirement levels’, RFC 2119.
http://www.ietf.org/rfc/rfc2119.txt

http://doi.org/10.5479/ADS/bib/2021ivoa.spec.0616C


http://doi.org/10.5479/ADS/bib/2012ivoa.spec.0827D

http://doi.org/10.5479/ADS/bib/2019ivoa.spec.1011D

http://doi.org/10.5479/ADS/bib/2021ivoa.spec.1102D

http://doi.org/10.5479/ADS/bib/2019ivoa.rept.0520D

http://doi.org/10.5479/ADS/bib/2017ivoa.spec.0517D
http://doi.org/10.5479/ADS/bib/2019ivoa.spec.0927D

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4486400/

https://ui.adsabs.harvard.edu/abs/2005mcp..book...83K

http://doi.org/10.5479/ADS/bib/2017ivoa.spec.0509L

http://www.schweizerbart.de//publications/detail/isbn/9783510651887/Strunz_Hugo__Nickel_Ernest_H_Strunz

http://doi.org/10.5479/ADS/bib/2008ivoa.spec.1030O

http://doi.org/10.5479/ADS/bib/2017ivoa.spec.0530P

http://doi.org/10.5479/ADS/bib/2006ivoa.spec.1101S