Conventions for structured data tables in Systems Biology

Wolfram Liebermeister[1], Timo Lubitz[2], and Jens Hahn[2]

[1] Institut für Biochemie, Charité - Universitätsmedizin Berlin
[2] Institut für Biophysik, Humboldt-Universität zu Berlin

Abstract

Data tables in the form of spreadsheets or delimited text files are the most utilised data format in Systems Biology. However, they are often not sufficiently structured and lack clear naming conventions that would be required for modelling. We propose the SBtab format as an attempt to establish an easy-to-use table format that is both flexible and clearly structured. It comprises defined table types for different kinds of data; syntax rules for usage of names, shortnames, and database identifiers used for annotation; and standardised formulae for reaction stoichiometries. Predefined table types can be used to define biochemical network models and the biochemical constants therein. The user can also define own table types, adjusting SBtab to other types of data. Software code, tools, and further information can be found at www.sbtab.net.

1 Introduction

Spreadsheets and delimited text tables are the most utilised data formats in Systems Biology. They are easy to use and can hold various types of data. Tables can not only store omics data, but also metabolic network models described by lists of biochemical reactions. However, when tables are exchanged within scientific collaborations, modellers usually prefer tables that can be processed automatically, and the flexibility of spreadsheets can become a disadvantage. If table structures and nomenclature vary from case to case, parsing becomes laborious and new files require new parsers. Furthermore, different naming conventions – for instance, for biochemical compounds – make it hard to combine data, for instance metabolic network models and omics data produced by different researchers. Therefore, rules for structuring tables and for consistent naming and annotations can make tables much more useful as exchange formats in Systems Biology collaborations and for usage in software tools. SBtab comprises a set of conventions for data tables that are supposed to make tables easier and safer to work with. We start with a couple of examples and then continue with a more formal specification of SBtab.

Example 1: A stoichiometric metabolic model A stoichiometric metabolic model can be defined by a list of biochemical reaction formulae, specifying the substrates, products, and their stoichiometric coefficients. Such reactions can be listed in a single column of a spreadsheet, and additional information may be provided: each reaction can have a number or identifier (defined only within the model) and can be linked to an entry in the database KEGG Reaction [1]. Furthermore, reactions may be catalysed by enzymes, which relates them to certain genes. All information could be stored in the following table:

Reaction	Sum formula	KEGG ID	Gene symbol
R1	ATP + F6P <=> ADP + F16P	R00658	pfk
R2	F16P + H2O <=> F6P + Pi	R01015	fbp

where ATP, F6P, ADP, F16P, H2O, and Pi are shortnames for metabolites to be used in the model. Although the information is complete and unambiguous, the parser still has to recognise that the columns Sum formula and KEGG ID contain reaction formulae and identifiers in certain formats. If the column names and the syntax of the reaction formulae vary from table to table (e.g. <-> is used instead of <=>), parsing becomes tedious. In the SBtab format, the table would look a little more complicated, but is easy to parse automatically:

!!SBtab	TableName='Reaction'	TableType='Reaction'	
!Reaction	!SumFormula	!Identifiers:kegg.reaction	!Gene:Symbol
R1	ATP + F6P <=> ADP + F16P	R00658	pfk
R2	F16P + H2O <=> F6P + Pi	R01015	fbp

In this table, elements highlighted by colours have special meanings (the colours themselves are just used in this text and are not part of the SBtab format). The SBtab table differs from the original table in several ways: the first line (starting with !!) declares that the table is an SBtab table of the type Reaction and must therefore satisfy syntax rules for this table type. The following line contains the column headers. They start with the ! character, emphasising that they were not chosen *ad hoc* by the user, but stem from a controlled vocabulary. The predefined column headers do not contain whitespaces. The header KEGG ID has been replaced by the term !Identifiers:kegg.reaction. This may look complicated, but it allows parsers to retrieve further data from databases in a database in a stable way[1] The syntax of the reaction formulae is also uniquely defined. In particular, the shortnames of metabolites must not contain any whitespaces or special characters, which simplifies parsing and makes them suitable as variable names for computer models. The meaning of these shortnames can be defined by providing standardised names or database identifiers in a second table of type Compound. The compound shortnames will then serve as keys to rows of this table.

!!SBtab	TableName='Compound'	TableType='Compound'
!Compound	!Name	!Identifiers:kegg.compound
F6P	Fructose 6-phosphate	C05345
ATP	ATP	C00002
ADP	ADP	C00008
F16P	Fructose 1,6-bisphosphate	C00354
H2O	Water	C00001
Pi	Inorganic phosphate	C00009
PEP	Phosphoenolpyruvate	C00074
AMP	AMP	C00020

Both tables together form an SBtab document describing a model. In practice, they can be stored as separate files, as sheets of a spreadsheet file, or within a single table. The following example contains all necessary information to build a stoichiometric model in the SBML (Systems Biology Markup Language) format [3]:

[1]The expression kegg.reaction is defined by the MIRIAM resources and used within SBtab. The URL of the KEGG database, defining the identifiers, may change in the future; however, KEGG's Miriam ID (provided by the the MIRIAM resources web service [2]) is guaranteed to remain stable in time.

!!SBtab	TableName='Reaction'	TableType='Reaction'	
!Reaction	!SumFormula	!Identifiers:kegg.reaction	!SBML:reaction:id
R1	ATP + F6P <=> ADP + F16P	R00658	r1
R2	F16P + H2O <=> F6P + Pi	R01015	r2
!!SBtab	TableName='Compound'	TableType='Compound'	
!Compound	!Name	!Identifiers:kegg.compound	!SBML:species:id
F6P	Fructose 6-phosphate	C05345	f6p
ATP	ATP	C00002	atp
ADP	ADP	C00008	adp
...	...	...	...

Here, we have added new identifiers (in the columns SBML:reaction:id and SBML:species:id) for Reaction and Compound entries to be used in SBML. Such extra names could be necessary if the original shortnames do not comply with SBML's rules for element identifiers.

Example 2: Kinetic constants In a second example, we specify numerical parameters, for example kinetic constants and metabolite concentrations that appear in a kinetic model. Each quantity can be related to a compound (e.g. a concentration), to a reaction (e.g. an equilibrium constant), or to several biological elements (e.g. to an enzyme and a compound, in the case of Michaelis-Menten constants). As in the previous example, these elements can be specified by unique identifiers, e.g. KEGG compound or reaction identifiers. Furthermore, each quantity has a value and a physical unit. In the SBtab format, we arrange this information in a table of type Quantity. Each row contains all information about one of the quantities:

!!SBtab	TableName='Quantity'	TableType='Quantity'			
!Quantity	!QuantityType	!Reaction:Identifiers:kegg.reaction	!Compound:Identifiers:kegg.compound	!Value	!Unit
keq_R1	equilibrium constant	R01061		0.156	dimensionless
kmc_R1_C1	Michaelis constant	R01061	C00003	0.96	mM
kic_R1_C1	inhibition constant	R01070	C00111	0.13	mM
con_C1	concentration		C00118	0.203	mM
...	...	...	...	...	...

The first two columns specify a name and a type for each quantity. The quantity types (substrate catalytic rate constant, equilibrium constant etc.) are not chosen *ad hoc*, but stem from the Systems Biology Ontology (SBO) [4]. This ensures a unique spelling and allows software to retrieve definitions and further information from the SBO web services. The biological elements (in this case, reactions, compounds, or both) are specified in the following two columns. Unnecessary fields remain empty. The column name Value – like some other mathematical terms – is defined for SBtab (arbitrary values in this example). Unit names are defined as in SBML (see below). If the table is used together with a metabolic model, we can use compound and reaction identifiers from the model instead of the Identifiers.org annotations [5]. In this case, the table would read:

!!SBtab	TableName='MyData'	TableType='Quantity'			
!Quantity	!QuantityType	!SBML:reaction:id	!SBML:species:id	!Value	!Unit
MyData_1	equilibrium constant	r1		0.156	dimensionless
MyData_2	Michaelis constant	r1	atp	0.96	mM
MyData_3	inhibition constant	r1	atp	0.13	mM
MyData_4	concentration		atp	1.5	mM
...	...	...	...	...	...

This table, together with a stoichiometric model and a choice of standard rate laws (like the modular rate

laws [6]) completely defines a kinetic metabolic model. The first column, if missing, would be automatically completed by the SBtab parser based on table type and row numbers.

2 The SBtab format

2.1 Basic conventions

SBtab comprises a list of conventions about the structure, nomenclature, syntax, and annotations in tables describing biochemical network models, kinetic parameters, and dynamic data. It contains

1. General rules for the **structure of tables** and the **syntax** used in table fields.
2. Defined **table types** for different kinds of information, each with possible **columns** with defined names and data types (see Table 1; An overview of all predefined table types and their possible columns is given in the appendix).
3. A **syntax for biochemical element annotations** pointing to databases or ontologies.
4. Rules for usage of **names**, **shortnames**, and **database identifiers** used for annotation.
5. **Naming rules for biochemical quantities** to specify the quantities, physical units, and mathematical terms (like `Mean` for mean values).
6. A syntax for **reaction sum formulae**.
7. A mechanism for **extending the format** by declaring new column or table types.

While the general rules apply to all kinds of data, the current version of SBtab is tailored for describing the structure of biochemical network models and the biochemical quantities therein. This is reflected by the table types defined in Table 1.

This specification for SBtab (version 0.8) introduces the general SBtab rules as well as formats and conventions for different types of use (see Section 2). It defines a list of table types (see Section 3) and explains the syntax of reaction formulae in the SBtab format (see Section 2.5). Finally, the specification references the available online tools for the handling of SBtab files (see Section 5) and includes an overview of all available SBtab table types in appendix A. Appendix B lists controlled vocabularies and database resources recommended to be used within SBtab.

Predefined terms and colour highlighting In the examples shown in this text, predefined SBtab entries are highlighted in colours. This is just for convenience and is not a part of the SBtab format. `Table types` and `Column types` defined by the SBtab format are listed in Table 1. `Shortnames` can be chosen *ad hoc* by the user; each of them needs to be defined by a table row. Shortnames have to be unique and consistent within a document, but may differ between documents. `Reserved names` are predefined in SBtab for recurrent mathematical expressions like "mean value". `Official names`, like the names used for databases, are defined by some other authority. `Free text` and other text including database IDs, numerical values, mathematical brackets, and operators is written in black.

2.2 SBtab tables and SBtab documents

General form of tables An SBtab document consists of several tables that refer to a common model or related data sets.

- All tables must use a common list of shortnames. For instance, a `Compound` table contains the column `!Compound`, and the elements from this column define compound shortnames to be used in the other tables.
- Several tables in a document may have the same type, but their table names (attribute `TableName`) must be unique.
- If a document is stored in a spreadsheet file, the sheets should be called like the table names. In character-separated text files (.csv or .tsv), a document can either be stored in several files with the filenames *basename_tablename.extension*, or tables are concatenated vertically, each preceded by a declaration row (starting with `!!`), and stored in a single table file.
- In character-separated files, irrespective of the extension (.csv or .tsv), it is assumed by default that the delimiters are tabulators. If other delimiters are used (comma or semicolon), this needs to be declared.
- If table cells contain special characters that are also used as cell delimiters (e.g. commas), the file must be provided in a form that excludes ambiguities (e.g. in the case of a comma-separated table containing commas with its fields, all cells must additionally be marked by quotation marks (`"..."`).

Declaration row and table attributes The top left field contains the table header, starting with `!!SBtab` and followed by the table attributes in the syntax *attribute name=´attribute value´*, separated by whitespaces. Mandatory attributes are `TableType` and `TableName`. Attributes that start with an exclamation mark `!` will be interpreted as if they marked the name of a column with all cells having the same content, the one given by the attribute value. The `ValueType` attribute can be used to define the meaning of `Value` columns, e.g. mean value, median value, etc (see Table 15). The `Unit` attribute can set a standard unit to be used in the table.

Column headers and definition table The second row contains the column headers. Columns whose headers start with a `!` are treated as SBtab columns and must adhere to the SBtab rules. Other columns can contain arbitrary content. SBtab has a number of predefined table types that can hold different kinds of data. Each table type has a number of mandatory or optional columns with specific properties. An overview is given below and in the appendix. However, users can also define their own table types and corresponding columns. This definition must be provided by the user in the form of a special `Definition` table. The default table (containing the predefined table and column types) is available on the SBtab website. The typical format of a `Definition` table is shown below.

!!SBtab	TableType='Definition'	TableName='Def'		
!Component	!ComponentType	!IsPartOf	!Format	!Description
SBML:reaction:id	Column	Reaction	String	SBML ID of reaction
SumFormula	Column	Reaction	String	Reaction sum formula
Enzyme	Column	Reaction	String	Enzyme catalysing the reaction
...	...	...	...	..

The `Format` column defines which type of entries a column can contain. Possibilities are `String`, `Shortname` (name of SBtab element, as defined in one of the SBtab tables), `Number` (integer or float in usual formats, or complex numbers like `1 + i 3`), or `Boolean` (with possible values 1 and 0, or `True` and `False`). More specific string formats (e.g., for reaction sum formulae) are currently not formally defined, but can be mentioned in the `Description` column.

Completeness To interpret the contents of a single table, other tables (e.g. describing shortnames) may be required. If a table does not require any other tables, we call it "complete". A document is complete if

Name	Contents	Usage
`Compound`	Names, IDs, properties of compounds	model structure
`Enzyme`	Names, properties of enzymes	model structure
`Protein`	Names, properties of proteins	model structure
`Gene`	Names, properties of genes	model structure
`Regulator`	Names, properties of gene regulators	model structure
`Compartment`	Names and IDs of compartments	model structure
`Reaction`	Chemical reactions	model structure
`Relationship`	Relations between different compounds	model structure
`Quantity`	Individual data for model parameters	quantitative data
`Definition`	Define custom column types, etc.	customise SBtab

Table 1: Overview of table types predefined in SBtab.

all names are defined, i.e. no unspecified information is required to interpret its contents. If a single table or a document are incomplete, the undefined names have to be known by the software, and an exchange with other software tools is likely to fail. If a table or document contains two elements, and there is no explicit information implying that they describe the same things, it is assumed that they describe different things.

2.3 Names of biochemical elements

Names and identifiers of model elements In the following, compounds, enzymes, genes, genetic regulators, and compartments will be called "biochemical entities". "Biochemical elements" comprises, in addition, reactions and biochemical quantities. Biochemical elements can be described by shortnames, official names, or database identifiers (IDs). The shortnames have to be declared within the SBtab document and have to satisfy syntactic rules. Each table starts with a column of the same name, containing the shortnames. Shortnames, the arbitrary element names used in a data set or model, must be unique, i.e. declared only once in a document; they must start with a letter or number and may not contain spaces or the special characters ":", ".". In columns containing database IDs, the column name (`!Identifiers:`*Identifiers*) specifies the database by a name (to be used in column names, IDs etc.) and an URI. We suggest to use preferably the databases listed in the Miriam file (see Table 16). Sometimes, elements may be characterised redundantly: e.g. the reaction catalysed by an enzyme, given in an `Enzyme` table, can be given by both shortname and database ID. In case of conflict, the information derived from the shortname (i.e. the database ID listed in the `Reaction` table) has higher priority.

Naming and specification of biological entities Tables of the types `Compound`, `Enzyme`, `Gene`, `Regulator`, or `Compartment` are called "entity tables". The biochemical meaning of the entities can be declared by different columns:

- `!Name` contains official names (it is good practice to use names from the suggested databases). Several names can be listed in one field, separated by "|". To declare from which database a name has been taken, the name can also be written as *DB*:*name*.
- `!Identifiers:`*Identifiers* contains IDs from a specified database. Annotations with database IDs follow the scheme defined by Identifiers.org (data collection and ID).

Localised compounds If a compound, enzyme, or genetic regulator is localised in a compartment, the corresponding localised entity can be denoted by *compound*[*compartment*] with square brackets, where

compound and *compartment* are the shortnames or IDs of the compound and the compartment used in the model. If a model contains several compartments, tools should treat the first compartment in the `Compartment` table as the standard compartment. The standard compartment will be used by default for all compounds that are not explicitly assigned to compartments.

2.4 Annotating biochemical elements with database identifiers

Biochemical elements are annotated with database IDs listed in special identifier columns. An `Identifiers` column contains annotations from one web resource, at most one annotation per element, and without qualifiers. The column item and the referenced ID are assumed to be linked by an "is" relationship (and not, for instance, "version of", which can exist in SBML annotations). A table can contain several `Identifiers` columns, which must refer to different data resources.

!!SBtab	TableName='Compound'	TableType='Compound'	
!Compound	!Identifiers:obo.chebi	!Identifiers:kegg.compound	...
water	CHEBI:15377	C00001	...
ATP	CHEBI:15422	C00002	...
phosphate	CHEBI:18367		...

To translate an element like `CHEBI:16865` into a valid Identifiers.org URI, `http://identifiers.org/` is concatenated with the data collection mentioned after `!Identifiers:` in the header (e.g. `obo.chebi`) and with the column item, separated by a slash[2]. For instance, the first annotation entry in the table above would be resolved to `http://identifiers.org/obo.chebi/CHEBI:15377`.

2.5 Syntax for reaction formulae

Chemical reactions can be described by reaction formulae (column `!SumFormula` in table `Reaction`; specifying the reactants, their stoichiometric coefficients, and possibly their localisation). The reaction arrow is denoted by `<=>`. Stoichiometric coefficients refer to substance amounts, not concentrations (this matters in the case of transport reactions). Stoichiometric coefficients of 1 are omitted; general stoichiometric coefficients, given by letters (e.g. `n`) are not allowed. If possible, the reaction formula should represent the actual stoichiometries experienced by the enzyme (i.e. `A <=> 2 B` rather than `0.5 A <=> B`). Substrates and products are given by shortnames, which must be defined in a `Compound` table. The order of substrates and the order of products are arbitrary; however, comparison of formulae is eased by using an alphabetical order. The localisation in compartments can be denoted as follows:

- Reaction in the default compartment: `A + 2 B <=> C + D`
- Transport reaction: `A[comp1] + 2 B[comp1] <=> C[comp2] + D[comp2]`

In the example, `A`, `B`, `C`, and `D` are compound shortnames, and `comp1` and `comp2` are compartment shortnames. The reversibility of reactions is not given by the sum formula, but by an extra column `!IsReversible` in the `Reaction` table.

[2] The elements from the column have to be translated into a URN-encoded form (as described in the URN specification): for instance, the colon in the identifier `CHEBI:16865` has to be replaced by the string "%3A" to create the URN `obo.chebi:CHEBI%3A16865`.

3 Predefined table types in SBtab

SBtab predefines a number of table types with specific properties. An overview is given in Table 1. The table types `Compound`, `Enzyme`, `Gene`, `Regulator`, `Compartment`, and `Reaction` describe model structures, the table types `Quantity` and `Relationship` are used for quantitative data.

3.1 Tables for biochemical network structures

As in example 1 (in the introduction section), biochemical networks consist of biochemical entities (e.g. metabolites or proteins) and reactions or interactions between them. The tables describing these entities (table types `Reaction`, `Compound`, `Compartment`, `Enzyme`, `Regulator`, and `Gene`) have to satisfy the following rules.

- **Entities** In tables describing biochemical entities (`Compound`, `Enzyme`, `Gene`, `Regulator`, `Compartment`), each row has to contain (i) a shortname as the primary key (in the column `!Compound`, `!Enzyme`, etc.) and (ii) at least one entry specifying the entity, like `!Name` or `!Identifiers:`*DB*. If a column shares the type of the table (e.g. a Compound column in a Compound table), it can be considered a primary key, that is, its elements should be unique and it should appear as the first column in the table. Optional columns - which may depend on the kinds of entities - are listed in Table A.3.
- **Reactions** A `Reaction` table lists chemical reactions, possibly with information about the corresponding enzymes, their kinetic laws, and their genetic regulation. It must contain at least one of the following columns: `!SumFormula`, `!Identifiers:`*DB*; optional columns are listed in Table 11. For an example, see example 1 in the introduction.
- **Enzymes, genes, and regulators** The connection between chemical reactions, the enzymes catalysing the reactions, and the genes coding for the enzymes can be complicated, but in many cases, there is a one-to-one relationship. In SBtab, there are different ways to express this relationship. Information about enzymes or genes and their regulation can be stored in a `Reaction` table if there is a one-to-one relationship between reactions, enzymes, and possibly genes. Otherwise, it is stored in separate tables `Enzyme` and `Gene` and the tables are interlinked *via* the columns `!Enzyme` (in table `Reaction`) and `!Gene` (in table `Enzyme`) or `!TargetReaction` (in an `Enzyme` table) and `!GeneProduct` (in a `Gene` table).

3.2 Tables for biochemical quantities

Numerical data (e.g. for time series or kinetic parameters) can be stored in tables and be linked to model elements via the latters' shortnames. There are two different table types for numerical data. Tables of type `Quantity` describe individual physical or biochemical quantities, for instance, kinetic parameters in a network model. These quantities can be linked to one entity, one reaction or enzyme, or both. If a quantity table contains several values for the same quantity, they appear in separate rows (for possible descriptions of provenance, see Table 10).

Tables of type `Quantity` describe single physical or biochemical quantities (e.g. individual kinetic constants). A quantity is defined by a type, a unit, possibly biochemical entities to which it refers, possibly a localisation, and possibly experimental or physical conditions. The columns contain the defining properties (e.g. unit, conditions, etc.) and their values. Quantities can refer to a compound, an enzyme or reaction, or a combination of them. For instance, a concentration refers to a substance, while a k^{M} value refers

to a metabolite and an enzyme. If there is a one-to-one relationship between reactions and enzymes, the k^{M} value can also be assigned to a compound/reaction pair or a compound/enzyme pair. Let us consider again example 2:

!!SBtab	TableName='Quantity'	TableType='Quantity'			
!Quantity	!QuantityType	!Reaction:Identifiers:kegg.reaction	!Compound:Identifiers:kegg.compound	!Value	!Unit
keq_R1	equilibrium constant	R01061		0.0984	dimensionless
kmc_R1_C1	Michaelis constant	R01061	C00003	0.96	mM
kic_R1_C1	inhibition constant	R01070	C00111	0.13	mM
con_C1	concentration		C00118	0.203	mM

To specify the parameters of a model, we refer to Reaction and Compound elements by shortnames rather than by resource IDs. In this form, the above example becomes

!!SBtab	TableName='Quantity'	TableType='Quantity'			
!Quantity	!SBO:Identifiers:obo.sbo	!Reaction	!Compound	!Value	!Unit
kcrf_R1	SBO:0000320	R1		200.0	1/s
keq_R1	SBO:0000281	R1		0.0984	dimensionless
kmc_R1_C1	SBO:0000027	R1	C1	0.96	mM
kic_R1_C2	SBO:0000261	R1	C2	0.13	mM
con_C3	SBO:0000196		C3	0.203	mM

This example also shows that quantity types can be specified by identifiers from the Systems Biology Ontology (SBO) in a column !SBO:Identifiers:obo.sbo. If all rows refer to the same type of quantity, then instead of specifying the type of quantity types in an !QuantityType column, we can also convert the quantity type directly into a column name, as in the following table:

!!SBtab	TableName='Quantity'	TableType='Quantity'	
!Quantity	!Reaction	!SBO:equilibrium constant	!Unit
keq_R1	R1	0.0984	dimensionless
keq_R2	R2	10	dimensionless
keq_R3	R3	100	dimensionless

This can be done for all measurement quantities defined in the SBO. As a header name, the quantity name (as defined in the Systems Biology Ontology) must be preceded by !SBO: (e.g. !SBO:equilibrium constant). The unit can also be added directly in square brackets, and mathematical quantities and scales (see Table 15 in appendix) can be added as in the following example (for median values of concentration in mM, expressed as decadic logarithms):

!!SBtab	TableName='Quantity'	TableType='Quantity'
!Quantity	!Compound	!SBO:concentration [mM:Log10]:Median
c_C1	C1	1
c_C2	C2	0
c_C3	C3	0.1

Other combinations (e.g., !SBO:concentration [mM] or !SBO:concentration:Median) are also possible. Standard units or mathematical values types can also be set by the table attributes ValueType and Unit in the declaration row.

A Quantity table can also store state-dependent quantities like concentrations, expression levels, or fluxes, like in the following example.

!!SBtab	TableName='Quantity'	TableType='Quantity'		
!Quantity	!Compound	!Condition	!SBO:concentration	!Unit
con_C1_wt	C1	wildtype	0.2	mM
con_C2_wt	C2	wildtype	1	mM
con_C3_wt	C3	wildtype	0.1	mM
con_C1_mu	C1	mutant	0.1	mM
con_C2_mu	C2	mutant	0.5	mM
con_C3_mu	C3	mutant	0.1	mM

3.3 Conversion to SBML models

SBML (Systems Biology Markup Language) models can be converted into SBtab documents and vice versa. By default, a given SBMl model will be converted into a document containing a Reaction, Compound, Compartment, and Quantity table. Likewise, such tables (or also a Reaction table alone) can be converted into an SBML (level 2) model.

In the conversion, `Compound` entries in SBtab correspond to `species` elements in SBML. By default, the entries in the `Compound` and `Reaction` columns are translated into `id` attributes of the SBML elements. If SBML IDs are directly specified within SBtab (in the columns `SBML:reaction:id`, `SBML:species:id`, `SBML:parameter:id`, `SBML:reaction:parameter:id`, etc), these will be used instead. Rate laws from the SBML code are stored in SBtab as strings within a `KineticLaw` column. `Regulator` entries in SBtab correspond to `modifier` elements in SBML; multiple regulators can be described by a regulation formula (in the `Regulator` column): regulators are separated by a "|" symbol, while the sign of regulation can be denoted by + or -. For an enzyme allosterically inhibited by ATP and activated by ADP, the formula reads `-ATP|+Pyruvate` or `ATP|ADP` where inhibition and activition remain unspecified. Rules and events in SBML are currently not supported and will not be extracted from a given SBML model. Annotations are partially extracted (currently at most one annotation per SBML element).

The entries of `Quantity` tables can be inserted into SBML models or be extracted from them. By default, SBtab quantities referring to a reaction will become local reaction parameters in SBML, while other quantities become global parameters. The element of the `!Quantity` column will be used as SBML element ID unless it is overridden by the (optional) column `!SBML:parameter:id` (for global parameters) or `!SBML:reaction:parameter:id` (for local reaction parameters). Naming conventions for kinetic constants are given in [6], supplementary material Table A.5. Quantities that describe initial species amounts, initial species concentrations, or compartment sizes will be translated into the corresponding element attributes.

4 Summary of SBtab rules

SBtab implements the following conventions.

- **Shortnames** Model elements (e.g. compounds) are referred to by shortnames, which are defined in the corresponding table (e.g. `Compound` for compounds) Shortnames must be unique within an SBtab document. The first column of each table shares the name of the table type (e.g. column `!Compound` in table type `Compound`) and contains the shortnames, which serve as primary keys for this table and must therefore be unique.
- **Order of columns** The allowed column types depend on the table type, but their order is arbitrary. The only exception is the first column, which contains the shortnames (acting as keys for this table)

and whose name corresponds to the table type. However, it is good practice to sort the columns by importance and to arrange related columns next to each other.

- **ASCII Characters** The table fields contain only plain text. The format is case-sensitive, but the choice of fonts (bold, italic) does not play a role. Double quotes should not be used.
- **Table types and column names** Table types and their possible columns are defined in appendix A. Column names may not contain any special characters or white spaces (parsers should ignore these characters).
- **Comment lines** Table lines starting with a "%" character contain comments and are ignored during parsing.
- **Comments and references** Additional information about table elements can be stored in the optional columns `!Comment`, `!Reference`, `!ReferencePubMed`, and `!ReferenceDOI`, which can appear in all tables.
- **Unrecognised table or columns** Columns with unknown headers (not starting with `!`), or unrecognised header starting with `!` may appear in SBtab tables. They can be used, but are not supported by the parser. The use of undefined columns is inadviseable.
- **Declaration line** The first line, starting with `!!SBtab` must declare at least the attributes: `TableType`, `TableName`, and possibly the properties `Version` (for versions of the table), `SBtabVersion` (for SBtab version used), and `Document`. The entries can be separated by whitespaces or be given in separate fields.
- **Identifiers** Identifiers for compounds, compartments etc. can be specified in columns with a header "*ElementType*:`Identifiers:`*DB*").
- **Missing elements** If an element is missing, the table field is left empty. Missing numerical values can also be indicated by non-numerical elements like `?` or `na` (for "not available"). Mandatory fields must not be empty.
- **Formulae** Reaction sum formulae must be written in a special format explained below.
- **Reserved names** In the SBtab format, there are reserved names for (i) table types (marked by colours in this text); (ii) column names; (iii) types of biological elements (see Table 17); and (iv) types of biochemical quantities or mathematical terms (e.g. `Mean`) for them (see Table 18), and physical units.
- **Physical units** In SBtab, it is recommended to use the units listed in the SBML specification (see sbml.org/Documents/Specifications)[3]. As good practice, derived units (e.g. `kJ/mol`) and reciprocal units (e.g. `1/s`) should be given in the simplest possible form, in necessary using multiplication, division, exponentials, and round brackets (e.g. `gram/m^3`).

5 SBtab online tools

To simplify the usage of SBtab, we provide several online tools at `www.sbtab.net`.

[3]The following units are supported by SBML: `ampere`, `gram`, `katal`, `metre`, `second`, `watt`, `becquerel`, `gray`, `kelvin`, `mole`, `siemens`, `weber`, `candela`, `henry`, `kilogram`, `newton`, `sievert`, `coulomb`, `hertz`, `Litre`, `ohm`, `steradian`, `dimensionless`, `item`, `lumen`, `pascal`, `tesla`, `farad`, `joule`, `Lux`, `radian`. Orders of magnitude can be denoted by `k`, `M`, `c`, `m`, `mu`, `n`, `p`, `f` for Kilo, Mega, Centi, Milli, Micro, Nano, Pico, Femto. If a parameter is dimensionless, it has to be annotated as `dimensionless`.

1. **An online validator for SBtab files.** The online validator tool checks whether SBtab files (in .csv or .xls format) adher to the SBtab conventions. If a problem is identified by the validator, an instruction on how to fix the problem is provided. The validation is based on a `Definition` table defining the possible table and column types. The following validation rules are applied:

   (a) Initially, it has to be verified that the given table format is supported.

   (b) Each table must contain a declaration line, including a table type definition. The declaration line is checked for correct syntax. Both table type and name must be declared in the declaration line. The table type needs to be recognised, otherwise the validator will stop the validation process.

   (c) The next row of the SBtab file is the header row that holds the column names. The first column is checked for consistence with the table type. The column headers are checked whether they are known predefined columns for this specific table type. If this is not the case, a warning is issued about the unknown column name.

   (d) It is checked whether the file contains data besides the declaration line and the header line. Furthermore, if there are empty data rows, a warning is issued. Every data row should have the same length (number of columns).

   (e) The row fields are checked whether their content has the correct data format referring to the SBtab definition file. The unique identifiers in the first column must not begin with '+' or '-'. It is good practise to use no special characters in the row identifiers (in the first column) aside from the underscore '_'.

2. **An online parser for SBtab.** The SBtab parser uses the python package `tablib` to import the SBtab file. The parser itself provides different functions for editing the data and for using them in Python directly:

   (a) Extracting table information (type, name, etc.) from the table.

   (b) Addition of rows and columns to the SBtab table.

   (c) Editing and export of the table content in rows, columns, and single entries. An export as a Python dictionary is also possible, to ensure easy access to the data for python programmers.

   (d) Switching of columns and rows in the table (matrix transposition). As some data are stored conveniently in transposed spreadsheets, some tables need to be transposed to have better access to its content.

3. **Online conversion tool of SBtab to and from SBML** The online conversion tool can create valid SBtab files from SBML models and vice versa. For the conversion from SBtab to SBML it has to be assured that at least an SBtab file of the table type `Reaction` is provided, since Reaction SBtabs contain the crucially required structure for an SBML model. As additional information, the following SBtab table types can be used for the conversion to SBML: `Compound`, `Compartment`, and `Quantity`. Biochemical annotations that are contained in the SBtab files correctly will be attached to the corresponding SBML elements. All information comprised in the aforementioned table formats can be converted to the SBML structure, as long as they are adhering to the correct SBtab syntax. Therefore, it is recommended to validate the SBtab files with the online validator before recruiting them for a conversion to SBML.
   The conversion from an SBML model file to SBtab produces four tables containing model information: The (i) `Reaction` SBtab contains a list of the reactions of the SBML file, including their sum formula, kinetic laws, irreversibility, annotations, and more. All species from the model can be found in the (ii) `Compound` SBtab. Their location, charge, annotations, and whether they are constant are

provided in the SBtab if they are part of the SBML file. Analogously, a (iii) `Compartment` holds all structural information of the cellular compartments. Finally, the (iv) `Quantity` SBtab file lists all parameters that are part of the model. Also their numerical values and units will be provided. It is of no importance if the parameters are global or local parameters in the SBML model. Their origin in this regard will be transferred to the SBtab file as well.

All SBtab files can be displayed online as HTML tables. If annotations are correctly provided, they will link to the online database by a mouse click. The SBML models that can be used as input and will be produced as output models follow the terms and conditions of SBML Level 2, Version 4.

Acknowledgements

The authors thank Dagmar Waltemath, Hans-Michael Kaltenbach, Dirk Wiesenthal, Jannis Uhlendorf, Anne Goelzer, Jörg Büscher, Avi Flamholz, Phillipp Schmidt, and Matthias König for contributing to this proposal, and Edda Klipp for financial support. This work was funded by the German Research Foundation (LI 1676/2-1), the European Commission (projects BaSysBio and UNICELLSYS), and the German Federal Ministry of Education and Research (project OncoPath).

References

[1] M. Kanehisa, S. Goto, S. Kawashima S, and A. Nakaya. The KEGG databases at genomenet. *Nucleic Acids Research*, 30:42–46, 2002.

[2] C. Laibe and N. Le Novère. MIRIAM Resources: tools to generate and resolve robust cross-references in Systems Biology. *BMC Systems Biology*, 1(1):58, 2007.

[3] M. Hucka, A. Finney, H.M. Sauro, H. Bolouri, J.C. Doyle, H. Kitano, A.P. Arkin, B.J. Bornstein, D. Bray, A. Cornish-Bowden, A.A. Cuellar, S. Dronov, E.D. Gilles, M. Ginkel, V. Gor, I.I. Goryanin, W.J. Hedley, T.J. Hodgman, J.H. Hofmeyr, P.J. Hunter, N.S. Juty, J.L. Kasberger, A. Kremling, U. Kummer, N. Le Novère, L.M. Loew, D. Lucio, P. Mendes, E. Minch, E.D. Mjolsness, Y. Nakayama, M.R. Nelson, P.F. Nielsen, T. Sakurada T J.C. Schaff, B.E. Shapiro, T.S. Shimizu, H.D. Spence, J. Stelling, K. Takahashi, M. Tomita, J. Wagner, J. Wang, and the SBML Forum. The Systems Biology Markup Language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[4] M. Courtot et al. Controlled vocabularies and semantics in systems biology. *Molecular Systems Biology*, 7(543), 2011.

[5] N. Juty, N. Le Novère, and C. Laibe. Identifiers.org and MIRIAM Registry: community resources to provide persistent identification. *Nucleic Acids Research*, 40:D580–D586, 2012.

[6] W. Liebermeister, J. Uhlendorf, and E. Klipp. Modular rate laws for enzymatic reactions: thermodynamics, elasticities, and implementation. *Bioinformatics*, 26(12):1528–1534, 2010.

[7] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

A Overview of table types

A.1 Document and table attributes

Table attributes

Name	Type	Format	Mandatory	Content
TableType	text	string	✓	Table type (as defined in definition table)
TableName	text	string	✓	Table name
ValueType	text	string		Value type (see Table 15)
Unit	text	string		Standard unit
Version	text	string		Table version (free text)
SBtabVersion	text	string		SBtab version number
Document	text	string		SBtab document name
ReferenceDescription	text	string		Name of reference description
Document	text	string		Document name
ReferenceCitation	text	string		Citation, unique identifier, unambiguous URL
ModelCreators	text	string		Name and contact information for model creators
ModelCreationTime	text	string		Date and time of model creation and last modification
TermsOfDistribution	text	string		Terms of distribution

Table 2: Possible table attributes (to appear in declaration row). The attributes in the lower part would be necessary for MIRIAM compliance. `ReferenceCitation` should also identify the authors of the model.

A.2 General column types

All table types

Name	Type	Format	Content
!Description	text	string	Description of the row element
!Comment	text	string	Comment
!ReferenceName	text	string	Reference title, authors, etc. (as free text)
!ReferencePubMed	text	string	Reference PubMed ID
!ReferenceDOI	text	string	Reference DOI

Table 3: Columns that can appear in all tables

All entity and reaction tables

Name	Type	Format	Content
!Name	text	string	Entity name
!Identifiers:*DataCollection*	resource ID	string	Entity ID
!MiriamAnnotations	annotation	string	Entity ID (JSON string)
!Type	text	string	Biochemical type of entity (examples see Table 17)
!Symbol	text	string	Short symbol (e.g., gene symbol)
!PositionX	number	float	x coordinate for graphical display
!PositionY	number	float	y coordinate for graphical display

Table 4: Columns that can appear in all entity (i.e. `Compound`, `Enzyme`, `Gene`, `Regulator`, and `Compartment`) and `Reaction` tables.

A.3 Predefined table types

`Compound`

Name	Type	Format	Content
`!Compound`	shortname	string	Compound shortname
`!SBML:species:id`	SBML element ID	string	SBML Species ID of the entity
`!SBML:speciestype:id`	SBML element ID	string	SBML SpeciesType ID of the entity
`!InitialValue`	number	float	Initial amount or concentration
`!Unit`	string	string	Unit for initial value
`!Location`	shortname	string	Compartment for localised entities
`!State`	shortname	string	State of the entity
`!CompoundSumFormula`	text	string	Chemical sum formula
`!StructureFormula`	text	string	Chemical structure formula
`!Charge`	number	integer	Electrical charge number
`!Mass`	number	float	Molecular mass
`!Unit`	text	string	Physical unit
`!IsConstant`	Boolean	Boolean	Substance with fixed concentrations
`!EnzymeRole`	shortname	string	Enzymatic activity
`!RegulatorRole`	shortname	string	Regulatory activity

Table 5: Columns that can appear in `Compound` tables

`Enzyme`

Name	Type	Format	Content
`!Enzyme`	shortname	string	Enzyme shortname
`!CatalysedReaction`	shortname	string	Catalysed reaction
`!KineticLaw:Name`	name	string	Rate law (name as in SBO)
`!KineticLaw:Identifiers.obo.sbo`	shortname	string	Rate law SBO identifier
`!Pathway`	text	string	Pathway name (free text)
`!Gene`	shortname	string	Gene coding for enzyme (shortname)
`!Gene:Name`	string	string	Gene coding for enzyme (name)
`!Gene:Symbol`	string	string	Gene coding for enzyme (short symbol)

Table 6: Columns that can appear in `Enzyme` tables

`Protein`

Name	Type	Format	Content
`!Protein`	shortname	string	Protein shortname
`!Name`	text	string	Protein name
`!Symbol`	string	string	Protein symbol
`!Gene`	shortname	string	Gene shortname
`!Gene:Name`	text	string	Gene name
`!Gene:Symbol`	string	string	Gene symbol
`!Gene:LocusName`	string	string	Gene locus name
`!Mass`	number	number	Protein mass
`!Size`	number	number	Protein size

Table 7: Columns that can appear in `Protein` tables

`Gene`

Name	Type	Format	Content
`!Gene`	shortname	string	Gene shortname
`!Name`	text	string	Gene name
`!Symbol`	string	string	Gene symbol
`!LocusName`	string	string	Gene locus name
`!GeneProduct`	shortname	string	Gene product shortname
`!GeneProduct:Name`	string	string	Gene product name
`!GeneProduct:Symbol`	string	string	Gene product symbol
`!GeneProduct:SBML:species:id`	SBML element ID	string	SBML ID of protein
`!Operon`	shortname	string	Operon in which gene is located

Table 8: Columns that can appear in `Gene` tables

`Regulator`

Name	Type	Format	Content
`!Regulator`	shortname	string	Regulator shortname
`!State`	shortname	string	State of the regulator
`!TargetGene`	shortname	string	Target gene
`!TargetOperon`	shortname	string	Target operon
`!TargetPromoter`	shortname	string	Target promoter

Table 9: Columns that can appear in `Regulator` tables

`Compartment`

Name	Type	Format	Content
`!Compartment`	shortname	string	Compartment shortname
`!Identifiers:obo.sbo`	shortname	string	Compartment SBO term
`!SBML:compartment:id`	SBML element ID	string	SBML Compartment ID
`!OuterCompartment`	shortname	string	Surrounding compartment (short)
`!OuterCompartment:Name`	string	string	Surrounding compartment (name)
`!OuterCompartment:SBML:compartment:id`	SBML element ID	string	Surrounding compartment
`!Size`	number	float	Compartment size
`!Unit`	text	string	Physical unit

Table 10: Columns that can appear in `Compartment` tables

`Reaction`

Name	Type	Format	Content
`!Reaction`	shortname	string	Reaction shortname
`!SBML:reaction:id`	SBML element ID	string	SBML Reaction ID
`!SumFormula`	SumFormula formula	string	Reaction sum formula
`!Location`	shortname	string	Compartment for localised reaction
`!Enzyme`	shortname	string	Enzyme catalysing the reaction
`!Model`	text	string	Model(s) in which reaction is involved
`!Pathway`	text	string	Pathway(s) in which reaction is involved
`!SubreactionOf`	shortname	string	Mark as subreaction of a (lumped) reaction
`!IsComplete`	Boolean	Boolean	Reaction formula includes all cofactors etc.
`!IsReversible`	Boolean	Boolean	Reaction should be treated as irreversible
`!IsInEquilibrium`	Boolean	Boolean	Reaction approximately in equilibrium
`!IsExchangeReaction`	Boolean	Boolean	Some reactants are left out
`!Flux`	number	float	Metabolic flux through the reaction
`!IsNonEnzymatic`	Boolean	Boolean	Non-catalysed reaction
`!KineticLaw:Name`	name	string	Rate law (name as in SBO)
`!KineticLaw:Identifiers.obo.sbo`	shortname	string	Rate law SBO identifier
`!Gene`	shortname	string	see table type `Enzyme`
`!Gene:Symbol`	string	string	see table type `Enzyme`
`!Operon`	shortname	string	see table type `Gene`
`!Enzyme:Name`	string	string	Name of enzyme
`!Enzyme:Identifiers:ec-code`	string	string	EC number of enzyme
`!Enzyme:SBML:species:id`	SBML element ID	string	SBML ID of enzyme
`!Enzyme:SBML:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!Enzyme:SBML:reaction:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!BuildReaction`	Boolean	Boolean	Includereaction in SBML model
`!BuildEnzyme`	Boolean	Boolean	Include enzyme in SBML model
`!BuildEnzymeProduction`	Boolean	Boolean	Describe enzyme production in SBML model

Table 11: Columns that can appear in `Reaction` tables. The lower section lists, again, column types from Table A.3.

`Relationship`

Name	Type	Format	Content
`!Relationship`	shortname	string	Type of quantitative relationship
`!From`	shortname	string	Element at beginning of arrow
`!To`	shortname	string	Element at arrowhead
`!IsSymmetric`	Boolean	Boolean	Flag indicating non-symmetric relationships
`!Value:`*QuantityType*	number	float	Numerical value assigned to the relationship

Table 12: Columns that can appear in `Relationship` tables.

`Quantity`

Name	Type	Format	Content
`!Quantity`	shortname	string	Quantity / SBML parameter ID
`!QuantityType`	shortname	string	Quantity type (e.g. from SBO)
ValueType	ValueType	string	Mathematical Term from table 15 (Mean, Std,...)
`!SBML:parameter:id`	SBML element ID	string	Parameter ID in SBML file
`!SBML:reaction:parameter:id`	SBML element ID	string	Parameter ID in SBML file
`!Unit`	text	string	Physical unit
`!Scale`	text	string	Scale (e.g. logarithm, see Table 15)
`!Provenance`	text	string	Name of data source (free text)
`!Condition`	text	string	experimental condition name (free text)
`!pH`	number	float	pH value in measurement
`!Temperature`	number	float	Temperature in measurement
`!Location`	shortname	string	Compartment (shortname)
`!Location:Name`	string	string	Compartment (name)
`!Location:SBML:compartment:id`	SBML element ID	string	SBML ID of compartment‘
`!Compound`	shortname	string	Related compound (shortname)
`!Compound:Name`	string	string	Related compound (name)
`!Compound:Identifiers:`*DataCollection*	resource ID	string	Compound ID
`!Compound:SBML:species:id`	SBML element ID	string	SBML ID of compound
`!Reaction`	shortname	string	Related reaction (shortname)
`!Reaction:Name`	string	string	Related reaction (name)
`!Reaction:Identifiers:`*DataCollection*	resource ID	string	Reaction ID
`!Reaction:SBML:reaction:id`	SBML element ID	string	SBML ID of reaction
`!Enzyme`	shortname	string	Related enzyme (shortname)
`!Enzyme:Name`	string	string	Related enzyme (name)
`!Enzyme:Identifiers:`*DataCollection*	resource ID	string	Enzyme ID
`!Enzyme:SBML:species:id`	SBML element ID	string	SBML ID of enzyme
`!Enzyme:SBML:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!Enzyme:SBML:reaction:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!Protein`	shortname	string	Related enzyme (shortname)
`!Protein:Name`	string	string	Related enzyme (name)
`!Protein:Identifiers:`*DataCollection*	resource ID	string	Protein ID
`!Protein:SBML:species:id`	SBML element ID	string	SBML ID of enzyme
`!Protein:SBML:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!Protein:SBML:reaction:parameter:id`	SBML element ID	string	SBML ID of enzyme
`!Gene`	shortname	string	Related gene
`!Organism`	shortname	string	Related organism

Table 13: Columns for numerical values and experimental conditions in tables of type `Quantity`.

`Definition`

Name	Type	Format	Content
`!TableType`	table type name	string	Table type for which column is defined
`!Column`	column name	string	Name of defined column
`!Format`	column entry format	string	type of element (e.g., `string`
`!Description`	text	string	Free text description

Table 14: Columns that can appear in `Definition` tables.

B Predefined terms and recommended controlled vocabularies

ValueType	Type	Format	Meaning
Value	number	float	Simple value
Mean	number	float	Algebraic mean
Std	number	float (positive)	Standard deviation
Min	number	float	Lower bound
Max	number	float	Upper bound
Median	number	float	Median
GeometricMean	number	float	Geometric mean
Sign	sign	{+,-,0}	Sign
ProbDist	Free text	string	Prob. distribution

Scale	Meaning
Lin	Linear scale (no transformation)
Ln	Natural logarithm
Log2	Dual logarithm
Log10	Decadic logarithm

Table 15: Terms for mathematical quantities and mathematical scales recommended for use in SBtab. Names of probability distributions can be, for instance, `Normal`, `Uniform`, `LogNormal`.

Database	MIRIAM URN	Contents	URI
SBO	obo.sbo	Quantities, rate laws	www.ebi.ac.uk/sbo/
CheBI	obo.chebi	Metabolites	www.ebi.ac.uk/chebi/
Enzyme nomenclature	ec-code	Enzymes	www.ebi.ac.uk/IntEnz/
KEGG Compound	kegg.compound	Compounds	www.genome.jp/KEGG/
KEGG Reaction	kegg.reaction	Reactions	www.genome.jp/KEGG/
KEGG Orthology	kegg.orthology	Genes	www.genome.jp/KEGG/
UniProt	uniprot	Proteins	www.uniprot.org/
SGD	sgd	Yeast gene loci	www.yeastgenome.org/
Gene Ontology	obo.go	Compartments	www.geneontology.org/
Taxonomy	taxonomy	Organisms	www.ncbi.nlm.nih.gov/Taxonomy/
SGD	sgd	Yeast proteins	www.yeastgenome.org/

Table 16: A selection of databases to be used in SBtab. For the complete list, see the MIRIAM resources [2].

Physical entity types	
protein complex	SBO:0000297
messenger RNA	SBO:0000278
ribonucleic acid	SBO:0000250
deoxyribonucleic acid	SBO:0000251
polypeptide chain	SBO:0000252
polysaccharide	SBO:0000249
metabolite	SBO:0000299
macromolecular complex	SBO:0000296

Compartments	
cell	GO:0005623
extracellular space	GO:0005615
membrane	GO:0001602
cytosol	GO:0005829
nucleus	GO:0005634
mitochondrion	GO:0005739

Table 17: Examples of biochemical entity types (with Systems Biology Ontology identifiers [4]) and cell compartments (with Gene Ontology identifiers [7]).

Name	SBO term	Default unit	Entities
standard Gibbs energy of formation	SBO:0000582	kJ/mol	Compound
standard Gibbs energy of reaction	SBO:0000583	kJ/mol	Compound
equilibrium constant	SBO:0000281	variable	Reaction
forward maximal velocity	SBO:0000324	mMol/s	Enzymatic reaction
reverse maximal velocity	SBO:0000325	mMol/s	Enzymatic reaction
substrate catalytic rate constant	SBO:0000321	1/s	Enzymatic reaction
product catalytic rate constant	SBO:0000320	1/s	Enzymatic reaction
Michaelis constant	SBO:0000027	mM	Enzyme, Compound
inhibitory constant	SBO:0000261	mM	Enzyme, Compound
activition constant	SBO:0000363	mM	Enzyme, Compound
Hill constant	SBO:0000190	dimensionless	Compound, Reaction
concentration	SBO:0000196	mM	Compound
biochemical potential	SBO:0000303	kJ/mol	Compound
standard biochemical potential	SBO:0000463	kJ/mol	Compound
rate of reaction (amount)	SBO:0000615	M/s	Reaction
rate of reaction (concentration)	SBO:0000614	mM/s	Reaction
Gibbs free energy of reaction	SBO:0000617	kJ/mol	Reaction
standard Gibbs free energy of formation	SBO:0000582	kJ/mol	Compound
standard Gibbs free energy of reaction	SBO:0000583	kJ/mol	Compound
transformed standard Gibbs free energy of reaction	SBO:0000620	kJ/mol	Reaction
transformed standard Gibbs free energy of formation	SBO:0000621	kJ/mol	Compound
transformed Gibbs free energy of reaction	SBO:0000622	kJ/mol	Reaction
thermodynamic temperature	SBO:0000147	K	Location (optional)
ionic strength	SBO:0000623	mM	Location (optional)
pH	SBO:0000304	dimensionless	Location (optional)

Table 18: A selection of quantity types to be used in SBtab in table types `Quantity`. The unit of equilibrium constants depends on the reaction stoichiometry. More quantities can be found in the Systems Biology Ontology [4].