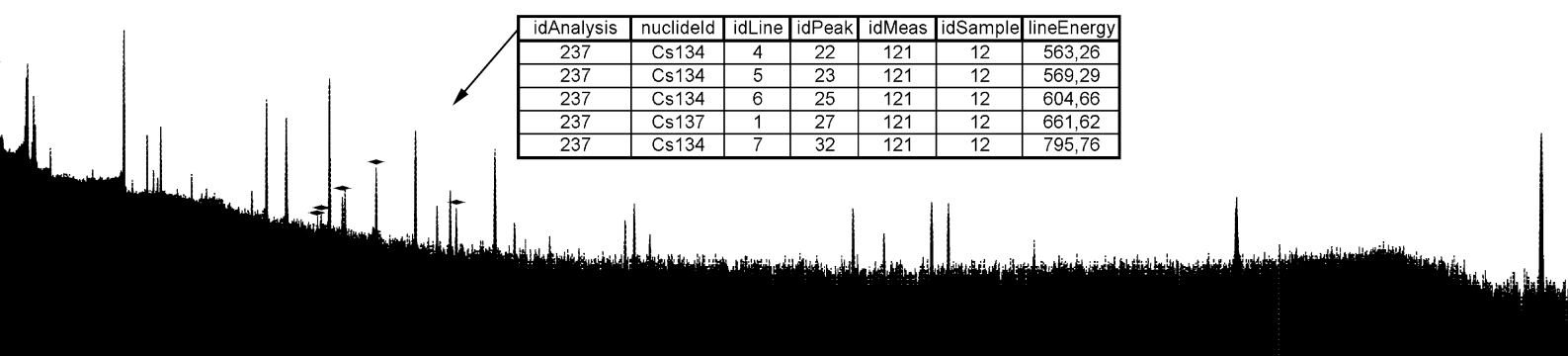


LINSSI
SQL DATABASE FOR GAMMA-RAY SPECTROMETRY
PART I: DATABASE
Version 1.1

Pertti Aarnio



idAnalysis	nuclideld	idLine	idPeak	idMeas	idSample	lineEnergy
237	Cs134	4	22	121	12	563,26
237	Cs134	5	23	121	12	569,29
237	Cs134	6	25	121	12	604,66
237	Cs137	1	27	121	12	661,62
237	Cs134	7	32	121	12	795,76



LINSSI
SQL DATABASE FOR GAMMA-RAY SPECTROMETRY
PART I: DATABASE
Version 1.1

Pertti Aarnio

Helsinki University of Technology
Department of Engineering Physics and Mathematics
Laboratory of Advanced Energy Systems

Teknillinen korkeakoulu
Teknillisen fysiikan ja matematiikan osasto
Energiateknologiat

Database and Software

© 2005, 2006

Pertti Aarnio¹, Jarmo Ala-Heikkilä¹, Arto Isolankila², Antero Kuusi², Mikael Moring², Mika Nikkinen², Teemu Siiskonen², Harri Toivonen², Kurt Ungar³, Weihua Zhang³

¹Helsinki University of Technology, Radiation Physics Group

²Finnish Radiation and Nuclear Safety Authority

³Health Canada, Radiation Protection Bureau

Manual

© 2004, 2006, 2007

Pertti Aarnio

Distribution:

Helsinki University of Technology

Laboratory of Advanced Energy Systems

P.O. Box 4100

FI-02015 TKK

ISBN 951-22-8148-8

ISSN 1456-3320

For the latest version of this manual see:

<http://linssi.hut.fi/radphys/linssi>

Information in this document is subject to change without notice and does not represent any commitment on the part of the authors. The software described in this document is furnished under a license agreement. The user may not copy the software on magnetic or optical tape, disk or any other medium, for any other purpose than the license holder's personal use.

Copyright

This database and accompanying software and written materials are products of copyright © owners and thereby protected by international copyright laws and treaties. You must keep the software package in strict confidence and treat it like any other copyrighted material. You may not copy the software or the written materials accompanying the software package except as explicitly allowed by the license. The use of the software package must be in strict adherence with the license.

License

License conditions are defined in a separate document that must be consulted.

Disclaimer of Responsibility for the Software

The program is provided "as is" without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. The authors do not warrant that the functions contained in the software will meet any requirements or that the operation of the software will be uninterrupted or error free.

In no event will the authors be liable for any damages, including any lost profits, lost savings, or other incidental or consequential damages arising out of the use or inability to use the program, even if authors' representative has been advised of the possibility of such damages, or for any claim by any other party.

MySQL is a trademark of MySQL AB. SHAMAN is a trademark of Baryon Oy. SAMPO, MICROSAMPO and SAMPO 90 are trademarks of Logion Oy. Other trademarks are the property of their respective owners.

Any data may be defined in one place only.

If data are defined in more places, they will diverge (it will not stay the same, if it ever was). If data are changed while not in one place only, you never know whether you changed every instance. However, you need a method (document control) that assures that all places where the changed data is referenced from are informed of any change. Relational databases use this principle. The same applies to software: every function should be defined only once (it would have made the millennium problem a piece of cake!).

Niels R. Malotaux

Contents

1	Introduction	1
1.1	History	1
1.2	The Database System	2
1.3	A Stroll between the Tables	4
1.4	Tables	6
1.5	Units and Default Values	6
2	Stations	9
2.1	Stations	9
2.2	Mobile Coordinates	11
2.3	Weather	13
3	Air Filter Samples	15
3.1	Samplers	16
3.2	Sampling State-of-Health Data	17
3.3	Air Filter Samples	18
4	Calibration Samples	21
4.1	Calibration Samples	21
4.2	Calibration Nuclides	23
4.3	Calibration Libraries	24
5	CTBT Laboratory Samples	27
5.1	CTBT Laboratory Samples	28
5.2	CTBT Transports	31
5.3	CTBT Recipients	32
5.4	CTBT Messages	33
5.5	CTBT Sample Tracking	35
6	Measurement	37
6.1	Samples	37
6.2	Splitting and Combining Samples	40
6.3	Sources	41
6.4	Detectors	43
6.5	Attenuators	44
6.6	Shields	45
6.7	Measurement Setups	46
6.8	Measurements	48

7	Calibration Data and Functions	51
7.1	Calibrations	51
7.2	Calibration Points	54
7.3	Calibration Preferences	55
7.4	Calibration Types	56
	7.4.1 Peak Shape Calibrations	56
	7.4.2 Energy Calibration	56
	7.4.3 Efficiency Calibration	56
7.5	Calibration Functions	56
	7.5.1 Predefined Functions	56
	7.5.2 MathML presentation of the functions	58
	7.5.3 MathML support in SHAMAN	58
8	Analysis	61
8.1	Analyses	61
8.2	Peaks	65
8.3	Line Associations	70
8.4	Nuclides and Their Activities	73
8.5	Activity Limits	78
8.6	Nuclide Ratios	80
8.7	Final Analysis Results	82
	Bibliography	83
A	Naming Conventions	85

Chapter 1

Introduction

1.1 History

Development of a database system to support the analyses of gamma-ray spectra from the International Monitoring System (IMS) for the verification of the Comprehensive Nuclear-Test-Ban Treaty (CTBT) was started at the Finnish National Data Centre (FiNDC), situated at the Finnish Nuclear and Radiation and Safety Authority (STUK), together with the Radiation Physics Group of Helsinki University of Technology (HUT) in summer 2002. Before that there had been an analysis pipeline for CTBT spectra in continuous operation for three years already. That pipeline, based on UNISAMPO [1] and SHAMAN [2] software, is still operational and has analyzed hundreds of thousands spectra. The storage and retrieval of spectra and results have been based on a hierarchical file structure and have been successful. However, the file structure is able to support only simple queries and it was clear from the beginning that, once a reasonably priced relational database system becomes available, it would be the correct way to handle the increasing amount of information. In the summer of 2002 the FiNDC made the decision to use MySQL [3] open source relational database.

As a first step the analysis results from UNISAMPO were stored in the database. This was accomplished in the fall of 2002. The database consisted of only seven tables: **Header**, **Collection**, **Acquisition**, **Peaks**, **Nuclides**, **MDA** and **Hypothesis**. This version was running at the FiNDC from October 2002 to June 2003. During its testing period it was decided to develop a more comprehensive database that would fulfil also the needs of a more general community performing gamma-ray spectrum analysis work.

In January 2003 the Laboratory of Airborne Radioactivity (ASL) of STUK joined in the collaboration and took the initiative to develop the database further. At the same time STUK and the Radiation Protection Bureau (Health Canada) started a close cooperation on issues related to the CTBT and environmental monitoring. This cooperation was established at the level of directors of the institutes taking the form of Memorandum of Understanding. The extended collaboration produced the first draft specifications in February 2003. These specifications, later named version 0.8, comprised of 22 tables with 342 fields.

In June 2003 STUK arranged a meeting on database specifications with Health Canada and HUT. The work culminated to a common view on the structure of the database and on the principle of open availability of the scripts for creating and managing the database. In this meeting the database was given a name *Linssi*, which may be considered as an acronym for a Linux system for spectral information. The first version carrying this name was the version 0.9 of June 2003 containing 22 tables and 365 fields. From this version onwards the full

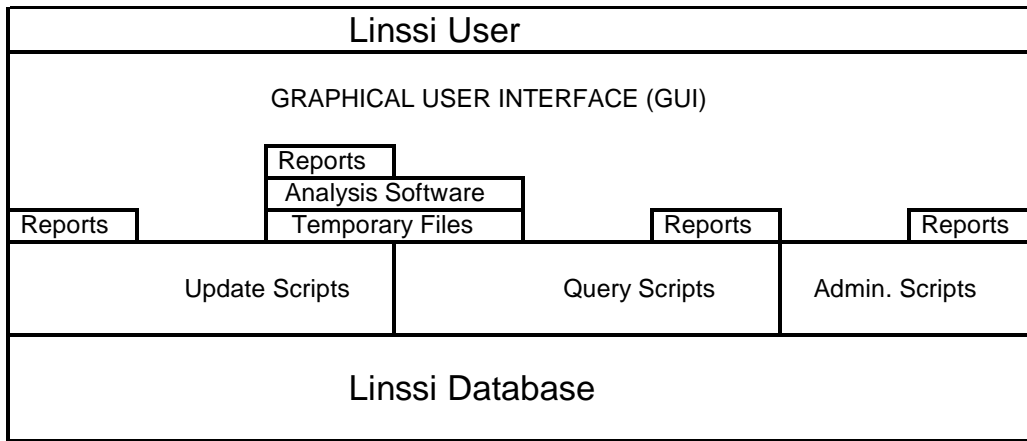


Figure 1.1: How *Linssi* interfaces with the user and analysis software.

UNISAMPO – SHAMAN pipeline was producing results to the database at the FiNDC. The version 0.9 operated from July to mid-November in 2003. *Linssi* v.1.0 β 6 was operational from mid-December to end of January 2004. The version 1.01 started operation at the FiNDC on February 9, 2004. *Linssi* v.1.01 contained 26 tables with 499 fields.

Already at the time of the release of the version 1.01 it was known that further extensions of *Linssi* were needed. The most important were the support of the CTBT laboratory samples and the bookkeeping of calibrations. Calibration tracking was achieved with a relatively minor but significant extension of the calibration tables. For calibration certificates a set of tables defining calibration samples was also added. An enhanced support of sample splitting and combining was also found necessary. Some streamlining and polishing was also carried out. *Linssi* version 1.1 was released on January 11, 2005. It contained 32 tables and 544 fields. After that CTBT tables were augmented. Because of some script compatibility issues and since the CTBT tables had been used only by the developers the version number was not changed. A new release of *Linssi* version 1.1 was published in July 7, 2006. It contains 32 tables and 557 fields.

1.2 The Database System

The *Linssi* database system consists of the database and the necessary scripts needed to do the updates and administer the database. An increasing number of query scripts will also be provided. The database is described in detail in this manual. The scripts provided with the database are mostly self documenting but are briefly described in the script manual [4].

The overall user environment as it is implemented at STUK is shown in Fig. 1.1, where the database and script layers form the *Linssi* distribution and the upper layers are commercial or in-house software. User interfaces with the system via a graphical user interface (GUI). The GUI to scripts is web-based and the analysis software has its own GUI. Most of the reports can be directly created using SQL queries but the analysis software has also a report generating possibility independent from *Linssi*. The analysis software, in our case UNISAMPO and SHAMAN, is interfaced to the update and query scripts using temporary files documented in the script manual. The amount of data processed in a laboratory performing gamma-ray spectrum analysis can be quite large and very heterogenous. It contains everything from the measured spectra to obscure notes on analysts' log books. Our intention has not been to

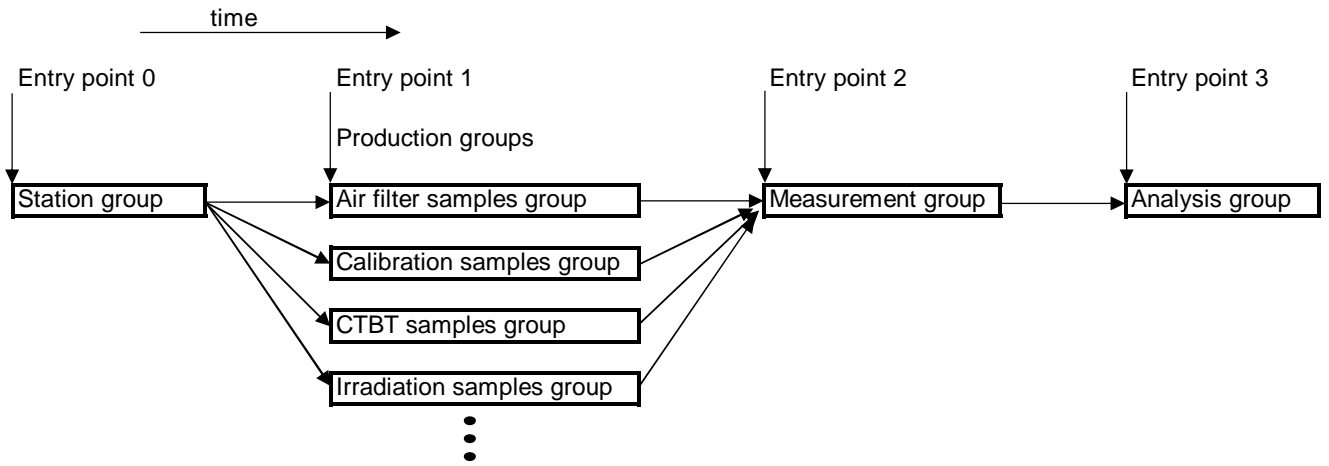


Figure 1.2: *Linssi* entry points and major table groups.

store everything in the *Linssi* database. However, the idea has been to include all the *relevant* information starting from the collection of radioactivity to a sample all the way to the final analysis results and conclusions made by the laboratory experts. Clear emphasis has been on the spectrum analysis results and on the information directly affecting the quality of these results. The information is meant to be complete enough to allow laboratory certification and thus also to facilitate outside review on the quality of the results based on the information available in the *Linssi* database. In case information not directly available in the database is needed, there should be enough pointers in *Linssi* to identify where the information might be available. In most cases that means information on the facility responsible for sample manufacturing and, possibly, responsible for its measurement. The number of fields in the analysis related tables is more than 200. That is quite a lot and we do not expect that they are all needed in every application. However, these are the generic fields readily available from our analysis software and with modern computers the overhead due to unused database fields is negligible.

We have made every effort to provide a database with information that is generic, i.e., not depending on the specific software used. An example of this type of information is nuclide activity. It does not depend on the analysis software used even though the results vary. On the other hand there exist tens of different peak shape models, for example. We have provided a relatively flexible model, but there certainly exist software for which our definitions do not apply without user modifications.

The *Linssi* design assumes three main entry points to the system (Fig. 1.2). They are sample production (**entry point 1**), sample measurement (**entry point 2**), and spectrum analysis (**entry point 3**). In addition there is the **entry point 0** denoting the station group of tables (Ch. 2), which can be updated relatively independently from the other table groups. In the time line of the analysis we assume that the stations have existed forever.

In **entry point 1** we start from the production or collection of activity to form the sample. That can be done in a multitude of ways. Currently (version 1.1) we have defined three different sample types, air filter samples (Ch. 3), calibration samples (Ch. 4) and CTBT laboratory samples (Ch. 5). The CTBT samples are quite specific and the calibration samples are just calibration samples. On the other hand, we believe that the air filter samples can be applied to other sample collection modes too. If that is not acceptable new production table groups can be defined. This is illustrated with the yet non-existent irradiation group of tables in the Fig. 1.2 above.

If we receive a sample to be measured we start from **entry point 2**, i.e., from preparing the source and measuring it (Ch. 6).

Finally, if we receive a gamma-ray spectrum for analysis, we start from **entry point 3**, i.e., send the spectrum directly to the analysis pipeline (Ch. 8).

If our laboratory controls the whole chain of events from **entry point 0** to **entry point 3** the database is updated as we go along. On the other hand, if the later entry points are applied the necessary data must be provided from outside and stored to the previous tables of the chain. Scripts are provided for each of these entry points. The entry points also define the most important keys in the *Linssi* database: `idSample` in table 6.1 **samples**, `idMeas` in table 6.8 **measurements**, and `idAnalysis` in table 8.1 identifying the sample, its measurements and its analyses, respectively.

1.3 A Stroll between the Tables

Let's take an introductory stroll between the tables in *Linssi*. The table lay-out is shown in Fig. 1.3. In the figure only the table names and key persons sitting in the table are shown. The arrows are drawn from the foreign keys to the primary keys they are referencing. Note that a key can simultaneously be both primary and foreign. The naming conventions of the keys and other fields are described in App. A.

As can be seen, almost half of the tables, tables 14 to 22, concentrate on gamma-ray spectrum analysis. The number of fields is also greatest in these tables. Spectrum measurement is covered by tables 7 to 13 and the sample production facility by tables 1, 2 and 4. The different sample production processes are covered in the tables of air filter samples group (Ch. 3), calibration samples group (Ch. 4), and CTBT lab samples group (Ch. 5).

Starting from table 1 we find the weather reigning outside the station, which itself is defined in table 2. The station is assumed to be the place where the radioactivity is collected to the sample or where the sample manufacturing takes place. Thus, by definition, all *in situ* measurements also take place at the station. However, nothing prevents defining, for example, a weather station that has nothing to do with sample production or measurement. If the station is mobile its positions are stored in table 4.

Obviously a sample can be produced in many different methods. Currently we have defined tables for calibration samples, air filter samples and CTBT laboratory samples. These groups will be described in the subsequent chapters.

After sampling the sample arrives to a measurement facility, which may, or may not, be situated at the sampling station. Table 7 characterizes the sample at this point. The sample is further processed into the actual source geometry, table 8, to be put on the spectrometer. From the sample it is possible to produce different sources, for example by using different containers. As long as the radionuclide content is not changed we are dealing with the same sample. It has just been transformed to a new source. Now the source is measured in a measurement setup described in table 12. The setup uses a detector, attenuator and shield described in tables 9, 10 and 11, respectively. Important information of the measurements themselves and, most importantly, the measurement results, i.e., gamma-ray-spectra are stored in table 13.

The measurements are followed by analyses. General information on analyses is stored in table 16. Analysis itself relies on calibrations of tables 14 and 15. The traditional gamma-ray spectrum peak analysis results are stored in table 17. They are followed by the identification and activity calculation results, tables 18 and 19, respectively.

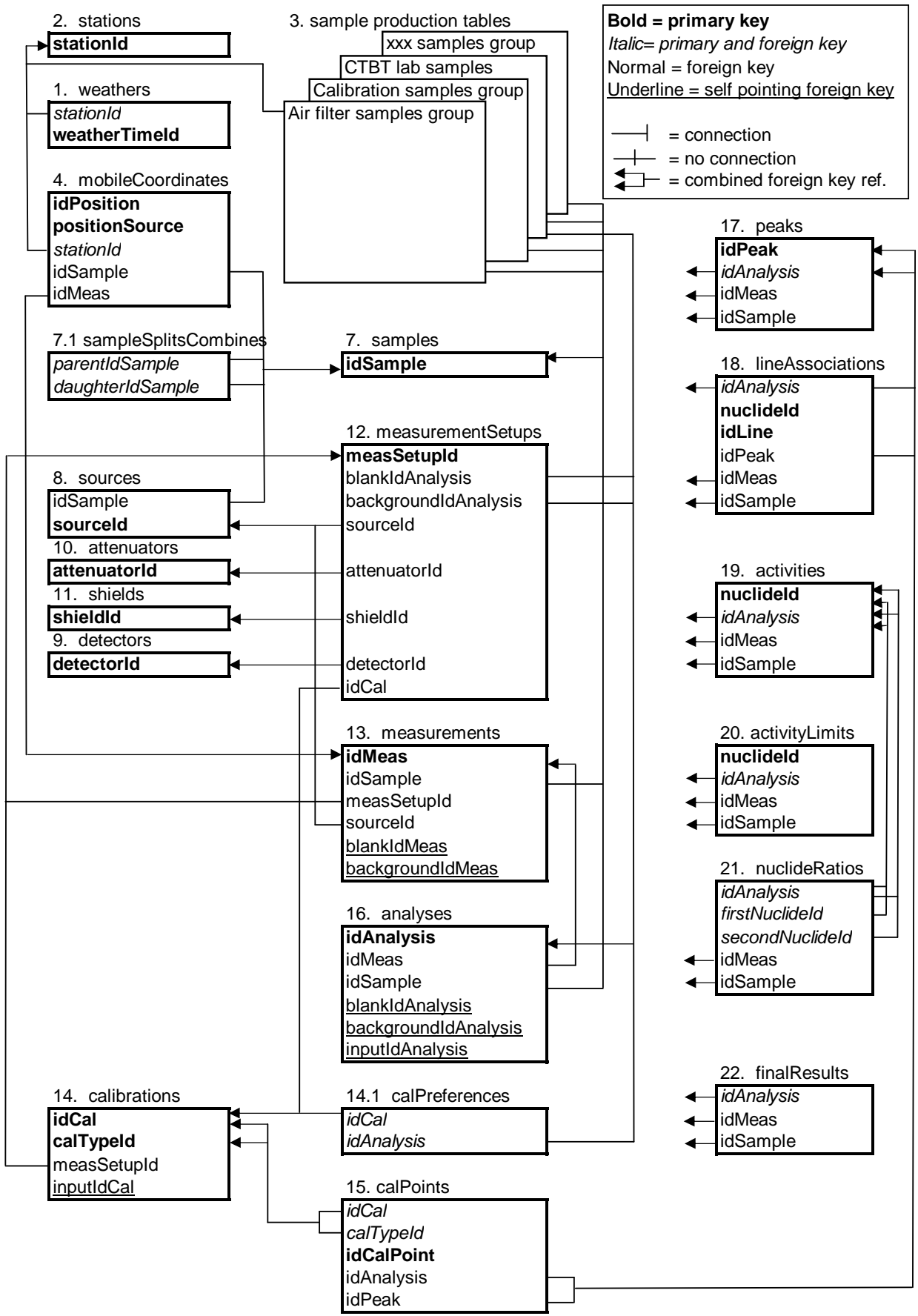


Figure 1.3: Table lay-out of the *Linssi* restaurant.

Minimum detectable activities and other activity limits are stored in table 20 and activity ratios for relevant nuclides in table 21. Finally the reviewed final results are stored in table 22. This table contains analysts' comments and pointers to the best analysis results.

It should be emphasized that a sample can be measured multiple times and that the resulting spectra can be also analyzed any number of times. The results of all measurements and analyses can be stored in the database.

1.4 Tables

In the following chapters each database table is described in detail. This is done in the form of a table followed by explanation of the independent fields. In the table header we show first the name of the database table. The column **Field** gives the name of the database column. The column **Type** defines the data type of the field i.e. whether it is character, integer, float, etc. The column **Length** gives the maximum size of the field in bytes. For variable size fields a zero is printed.

The column **Flags** gives information of the relational status of the field. The available flags are:

- P Primary key or a component of the primary key.
- A The field is automatically incremented every time a new entry is made.
- F Foreign key or a component of the foreign key.
- # Number (1,2,...). Foreign keys with the same number form a combined foreign key.
- S Self-referencing key that points to a record in the same table.
- U The field content must be unique within this table.
- I The field is indexed for faster reference.
- N The field must contain a value, i.e., it must not be NULL.

Note that the primary key may be composed of multiple fields. In that case, all included fields are marked as primary. A set forming the complete primary key must be unique. Primary keys are always indexed. Their components may also be indexed. Since there may be many foreign keys in a table, flag F is not enough to define a combined foreign key. For that purpose each combined foreign key is numbered, flag #.

1.5 Units and Default Values

We use only SI units and units outside the SI that are accepted for use with the SI, and thus consistent with the recommendations of the International Committee for Weights and Measures (CIPM, Comité International des Poids et Mesures) [5]. It is strongly recommended that users of *Linssi* follow the unit conventions of this manual. If different units are needed locally, the conversion should be performed in the interfacing scripts.

All dates and times (`datetime`, `timestamp`) are in Universal Time Coordinated (UTC).

The uncertainties in *Linssi* are given as one sigma absolute uncertainties. If, for example, uncertainty per cent is needed it should be calculated from the value and its absolute uncertainty when reading the database.

Linssi implementation does not include default values to fields. The action to be taken due to unknown field values should always be left to the user, i.e., to the software or analyst inputting or outputting data to or from *Linssi*. To facilitate this practice unknown values

should always be set `null`. That is automatically true for fields that are not filled, i.e. have been left empty. Note, however, that there are some exceptions. They are marked with flag `N`.

Chapter 2

Stations

The station group of tables of *Linssi* establishes the **entry point 0** of the database. Sample production takes place at the station, where the sample manufacturing equipments are situated. Nothing, however, prevents from also defining other types of stations. They may include a moving laboratory with samplers and measuring facilities, or even a simple weather station, for example.

Since the group aims to support any kind of station, it is quite generic. Basically a station has only a name (`stationId`), purpose (`stationType`) and location (`location`). It may be mobile and then its position is given in table `mobileCoordinates`. The third table of the group (`weathers`) gives the weather at the station.

2.1 Stations

Table 2.1: Stations

stations			
Field	Type	Length	Flags
<code>stationId</code>	varchar	40	PN
<code>stationType</code>	varchar	20	
<code>isMobile</code>	Boolean	1	
<code>address</code>	varchar	40	
<code>telephone</code>	varchar	40	
<code>location</code>	varchar	40	
<code>longitude</code>	double	8	
<code>latitude</code>	double	8	
<code>altitude</code>	double	8	
<code>comments</code>	text	0	

end of table.

`stationId` Name of the station.
`stationType` Type of the station. This is very application dependent. May be, for example, P for particulate station, `weather` for weather station, etc.

<code>isMobile</code>	TRUE if the station is mobile. In that case the coordinates of the station can be found in table 2.2 mobileCoordinates .
<code>address</code>	Address of the station.
<code>telephone</code>	International telephone number of the station.
<code>location</code>	Location of the station.
<code>longitude</code>	Longitude of the stationary station.
<code>latitude</code>	Latitude of the stationary station.
<code>altitude</code>	Altitude of the stationary station.
<code>comments</code>	Comments in English [en].

2.2 Mobile Coordinates

Table 2.2: Mobile coordinates of stations

mobileCoordinates			
Field	Type	Length	Flags
<code>idPosition</code>	int	4	PN
<code>positionSource</code>	varchar	20	PN
<code>stationId</code>	varchar	40	PN
<code>idSample</code>	int	4	F
<code>idMeas</code>	int	4	F
<code>positionType</code>	varchar	10	
<code>positionTime</code>	datetime	8	
<code>mission</code>	varchar	40	
<code>coordSystem</code>	varchar	20	
<code>speed</code>	double	8	
<code>heading</code>	double	8	
<code>numSatellites</code>	int	4	
<code>xCoordinate</code>	double	8	
<code>yCoordinate</code>	double	8	
<code>altitude</code>	double	8	
<code>hDOP</code>	double	8	
<code>vDOP</code>	double	8	
<code>pDOP</code>	double	8	
<code>age</code>	double	8	
<code>fix</code>	int	4	
<code>comments</code>	text	0	
(idSample) → samples(idSample)			
(idMeas) → measurements(idMeas)			

end of table.

This table tracks the coordinates of mobile stations, `stationId`. To facilitate the use of the primary key `idPosition` as a marker for identical positions from multiple positional devices the third primary key `positionSource` has been provided. And, to facilitate easy retrieving of the positions of sample collection and measurements, the foreign keys `idSample` and `idMeas` have been included.

In practice the station coordinates are continuously stored from GPS receiver(s). Asynchronously with the storing, sampling and measuring scripts retrieve data from the table, process them, and store the relevant positions back to the table. While processing they set their respective fields `idMeas` or `idSample` to correct values. A route processing script is additionally used to replicate the coordinates it deems relevant and identify them by setting `positionType` to `route`. After the mission the coordinates for which `idMeas = idSample = positionType = null` can be deleted.

<code>idPosition</code>	Position identifier. <i>Note 1:</i> Since <code>idPosition</code> is only a part of the combined primary key, it does not need to be unique. This can be useful when postprocessing of records in <code>mobileCoordinates</code> is needed. The postprocessor may store a modified record back to <code>mobileCoordinates</code> table with identical <code>idPosition</code> and <code>stationId</code> but with differing <code>positionSource</code> . <i>Note 2:</i> We recommend the use of Unix timestamp as the position identifier.
<code>positionSource</code>	Name of the source providing the coordinates. The station may have multiple GPS navigators, for example, or the source may be the postprocessor mentioned above.
<code>stationId</code>	See table 2.1 stations.
<code>idSample</code>	See table 6.1 samples.
<code>idMeas</code>	See table 6.8 measurements.
<code>positionType</code>	The following position types are defined: null: If both <code>idMeas</code> and <code>idSample</code> are <code>null</code> this is a raw unprocessed position. If not, this is a measurement or sampling position. route: A processed and validated point on station's route. gap: Before this position there is a gap in position information. GPS may have been off or lost its connection with satellites, for example.
<code>positionTime</code>	Date and time of the coordinate information.
<code>mission</code>	Name of the mission.
<code>coordSystem</code>	Name of the coordinate system in use.
<code>speed</code>	Station speed in meters per second [m/s].
<code>heading</code>	Direction the station is heading in degrees [°](0...360), 0° = North.
<code>numSatellites</code>	Number of satellites the GPS navigator sees.
<code>xCoordinate</code>	x-coordinate or longitude of the mobile station.
<code>yCoordinate</code>	y-coordinate or latitude of the mobile station.
<code>altitude</code>	Altitude of the mobile station.
<code>hDOP</code>	Precision class of dilution of precision of horizontal coordinates.
<code>vDOP</code>	Precision class of dilution of precision of vertical coordinate.
<code>pDOP</code>	Precision class of dilution of precision of position.
<code>age</code>	Age of the information in seconds [s].
<code>fix</code>	The method of fixing the position: 0 = invalid, 1 = GPS, 2 = DGPS, 3 = PPS, 4 = RTK, 5 = float RTK, 6 = estimated, 7 = manual, 8 = simulation.
<code>comments</code>	Comments in English [en].

2.3 Weather

Table 2.3: Weather at the station

weathers			
Field	Type	Length	Flags
stationId	varchar	40	PFN
weatherStartId	datetime	8	PIN
weatherEnd	datetime	8	
windDir	double	8	
windSpeed	double	8	
avgTemperature	double	8	
lowTemperature	double	8	
highTemperature	double	8	
snowfall	double	8	
rainfall	double	8	
humidity	double	8	
pressure	double	8	
stabilityClass	char	1	
comments	text	0	
(stationId) → stations(stationId)			

end of table.

Table **weathers** contains the information about the weather conditions at the station. The most important weather parameters can be continuously monitored. The table is preferably filled on-line even though that is not required. The average values are the averages over the report period.

stationId	See table 2.1 stations .
weatherStartId	The starting date and time of the weather report period.
weatherEnd	The ending date and time of the weather report period.
windDir	Average wind direction in degrees [°](0...360), 0° = North.
windSpeed	Average wind speed in meters per second [m/s]
avgTemperature	Average temperature in degrees centigrade [°C].
lowTemperature	Lowest temperature in degrees centigrade [°C] during the report period.
highTemperature	Highest temperature in degrees centigrade [°C] during the report period.
snowfall	Amount of snowfall in millimeters [mm] in the report period.
rainfall	Amount of rainfall in millimeters [mm] in the report period.
humidity	Average relative humidity of air in percent [%].
pressure	Average air pressure in hectopascals [hPa].
stabilityClass	Weather stability class.
comments	Comments in English [en].

Chapter 3

Air Filter Samples

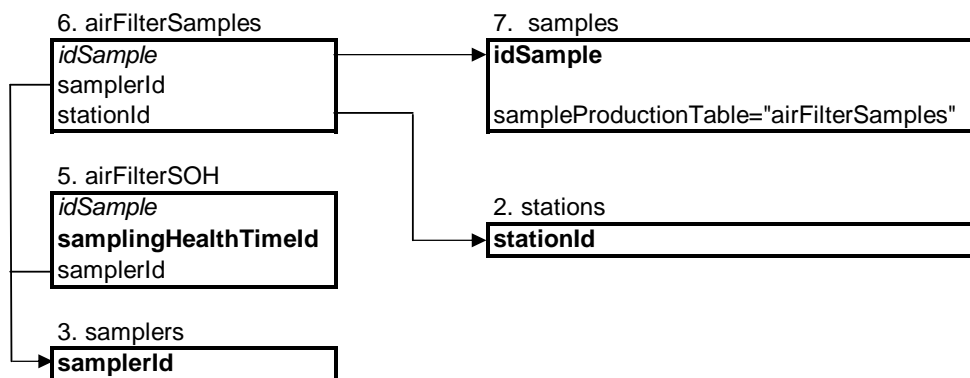


Figure 3.1: Tables of the air filter sample production group (3, 5, and 6) and their connection to other *Linssi* tables (2 and 7).

Air filter sampling is covered by the tables of the air filter samples group, Fig. 3.1. The three tables, 3, 5, and 6, define the **entry point 1**. The table `samplers` defines the ‘static’ properties of the sampler. The table `airFilterSamples` gives the information of the sampling specific to a given sample. Finally, the table `airFilterSOH` contains information of the State-Of-Health of the sampler as a function of time. This information is continuous in nature, i.e., not specific to a sample even though `idSample` is provided in the table in order to facilitate on-line following of its sampling.

In generic terms these three tables are used to describe the sample manufacturing process, i.e., the process whose result is a radioactive sample. As such they can be applied also to other type of samples. These may include irradiated samples, wipe samples, soil samples, xenon gas samples, etc. The specific activity production process, which here is air filter sampling, is denoted by the field `sampleProductionTable` in table 6.1 `samples`. For the time being the following `sampleProductionTable`s are supported: `calibrationSamples`, `airFilterSamples` and `ctbtLabSamples`. If the specific tables for your application are not available, it is recommended to use the tables for air filter samples. If that is out of the question, one should start from the **entry point 2**, i.e., from table 6.1 `samples`.

The terminology used, also in other than these three tables, reflects somewhat the first application of this database. We have, however, tried to minimize air sampling specific terminology and, where it is used, its interpretation in the context of other sample types should be obvious.

Calibration samples are discussed in Ch. 4 and CTBT laboratory samples in Ch. 5.

3.1 Samplers

Table 3.1: Samplers

samplers			
Field	Type	Length	Flags
samplerId	varchar	40	PN
stationName	varchar	40	
samplerType	varchar	20	
flowFactor1	double	8	
flowPower1	double	8	
flowFactor2	double	8	
flowPower2	double	8	
lastMaintenance	datetime	8	
nextMaintenance	datetime	8	
documentDir	varchar	255	
comments	text	0	

end of table.

Samplers consist of two air channels, 1 and 2, where total flow is obtained as a sum of these channels. In both channels there is a flow meter. The flow meters are described by two parameters flow factor F and flow power c . These parameters can be used for orifice plates, flow-nozzles or multi-orifice probes, or with any device for which two parameters are sufficient, only the method of flow rate calculation will differ. In our application the flow rate, f , is calculated from $f = F \times p^c$, where F is the flow factor in cubic meters per hour [m³/h], c is the flow power, and p is the pressure difference over the measuring device in pascals [Pa]. For p See also table 3.3 `airFilterSamples`.

<code>samplerId</code>	Name of the sampler.
<code>stationName</code>	This is the name of the station where the sampler is currently situated. This name should be identical to the <code>stationId</code> of the station, if known (see table 2.1 <code>stations</code>). <i>Note:</i> according to the naming conventions of App. A, this should be a foreign key with the name <code>stationId</code> , i.e., not <code>stationName</code> . However, since the sampler may have been moved to a station not having an entry in the table <code>stations</code> , the naming convention cannot be enforced.
<code>samplerType</code>	Name of the sampler type.
<code>flowFactor1</code>	Flow factor in cubic meters per hour [m ³ /h] for flow meter in channel 1.
<code>flowPower1</code>	Flow power of flow meter in channel 1.
<code>flowFactor2</code>	Flow factor in cubic meters per hour [m ³ /h] for flow meter in channel 2.
<code>flowPower2</code>	Flow power of flow meter in channel 2.
<code>lastMaintenance</code>	Date and time of the last maintenance of this sampler.
<code>nextMaintenance</code>	Scheduled date and time of the next maintenance of this sampler.
<code>documentDir</code>	The directory, or URI (Universal Resource Identifier), where the documentation of the sampler can be found.
<code>comments</code>	Comments in English [en].

3.2 Sampling State-of-Health Data

Table 3.2: State of health data (SOH) for air filter sampling process

airFilterSOH			
Field	Type	Length	Flags
idSample	int	4	PIFN
samplingHealthTimeId	datetime	8	PN
samplerId	varchar	40	F
pressDiff1	double	8	
airVolume1	double	8	
pressDiff2	double	8	
airVolume2	double	8	
flowRate	double	8	
correctedToNTP	Boolean	1	
comments	text	0	

(idSample) → samples(idSample)
 (samplerId) → samplers(samplerId)

end of table.

idSample	See table 6.1 samples.
samplingHealthTimeId	Date and time of SOH-data.
samplerId	See table 3.1 samplers.
pressDiff1	Air pressure difference in pascals [Pa] over the flow rate meter in channel 1.
airVolume1	Total volume of air sampled in channel 1 since collStart in cubic meters [m ³] at this moment.
pressDiff2	Air pressure difference in pascals [Pa] over the flow rate meter in channel 2.
airVolume2	Total volume of air sampled in channel 2 since collStart in cubic meters [m ³] at this moment.
flowRate	Air flow rate in [m ³ /h] at this moment.
correctedToNTP	If TRUE, air volumes and flow rate are corrected to Normal Temperature and Pressure (NTP), i.e., corrected to 20°C and 101.325 hPa.
comments	Comments in English [en].

3.3 Air Filter Samples

Table 3.3: Air filter samples

airFilterSamples			
Field	Type	Length	Flags
idSample	int	4	PFN
samplerId	varchar	40	F
stationId	varchar	40	F
collStart	datetime	8	
collEnd	datetime	8	
collTime	double	8	
samplerOnTime	double	8	
airVolumeTotal	double	8	
airVolume1	double	8	
presDiff1Start	double	8	
presDiff1End	double	8	
airVolume2	double	8	
presDiff2Start	double	8	
presDiff2End	double	8	
sampleSent	datetime	8	
correctedToNTP	Boolean	1	
comments	text	0	
(idSample) → samples(idSample) (samplerId) → samplers(samplerId) (stationId) → stations(stationId)			

end of table.

In this table, fields have been reserved for two channels in a sampler. This gives a possibility to use two types of filters simultaneously, typically active carbon and fiber. See also table 3.1 samplers.

idSample	See table 6.1 samples.
samplerId	This is the identifier of the sampler used. See table 3.1 samplers.
stationId	This is the identifier of the station where the sampling has been performed. See table 2.1 stations.
collStart	Date and time of the start of collection.
collEnd	Date and time of the end of collection.
collTime	Collection time in seconds [s].
samplerOnTime	Time the sampler was on during collection in seconds [s].
airVolumeTotal	Total air volume sampled through the filter(s) in cubic meters [m ³].
airVolume1	Air volume sampled through channel 1 in cubic meters [m ³].
presDiff1Start	Air pressure difference in pascals [Pa] over the flow rate meter in channel 1 at collStart.
presDiff1End	Air pressure difference in pascals [Pa] over the flow rate meter in channel 1 at collEnd.
airVolume2	Air volume sampled through channel 2 in cubic meters [m ³].

`presDiff2Start` Air pressure difference in pascals [Pa] over the flow rate meter in channel 2 at `collStart`.

`presDiff2End` Air pressure difference in pascals [Pa] over the flow rate meter in channel 2 at `collEnd`.

`sampleSent` Date and time the sample was sent for measurement.

`correctedToNTP` If TRUE, air volumes are corrected to Normal Temperature and Pressure (NTP), i.e., corrected to 20°C and 101,325.0 Pa.

`comments` Comments in English [en].

Chapter 4

Calibration Samples

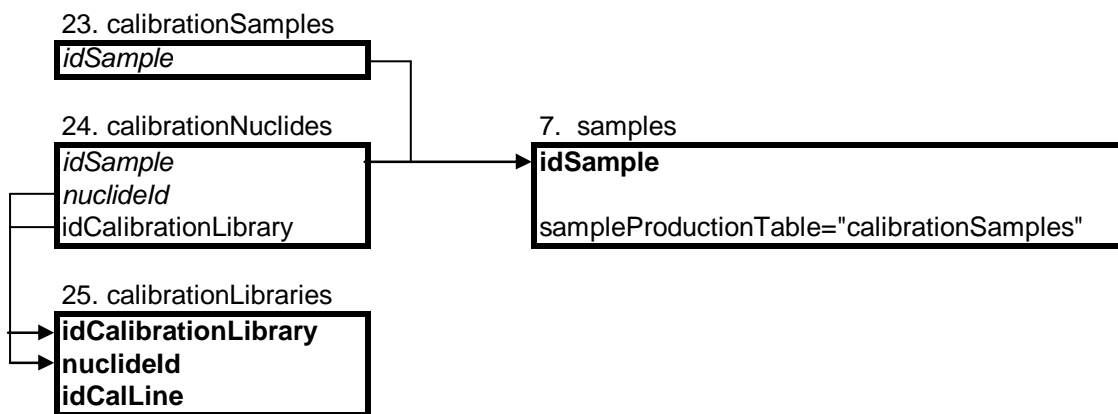


Figure 4.1: Tables of the calibration sample production group (23, 24, and 25) and their connection to other *Linssi* tables (7).

Calibration samples are identified with `sampleProductionTable` equal to `calibrationSamples` in table 6.1 `samples`. Calibration samples are samples with known activity and they are preferably certified. The certification is given in the table `calibrationSamples`, the information of the actual nuclides in the sample is given in the table `calibrationNuclides` and the specific nuclear data in the table `calibrationLibraries`.

4.1 Calibration Samples

Table 4.1: Calibration samples

calibrationSamples			
Field	Type	Length	Flags
<code>idSample</code>	<code>int</code>	4	PFN
<code>certificateNumber</code>	<code>varchar</code>	20	
<code>productCode</code>	<code>varchar</code>	20	
<code>sourceNumber</code>	<code>varchar</code>	20	
<code>manufacturer</code>	<code>varchar</code>	40	

continued on next page ...

... continued from previous page

Field	Type	Length	Flags
receiptTime	datetime	8	
overallAct	double	8	
comments	text	0	
(idSample) → samples(idSample)			

end of table.

idSample	Identification number of the calibration sample. See table 6.1 samples.
certificateNumber	Certificate number of the calibration sample manufacturer.
productCode	Manufacturer product code of the calibration sample.
sourceNumber	Manufacturer serial number of the calibration sample.
manufacturer	Name of the manufacturer.
receiptTime	Receipt date and time of the calibration sample.
overallAct	Overall activity of the calibration sample at receiptTime. This value is NOT certified.
comments	Comments in English [en].

4.2 Calibration Nuclides

Table 4.2: Calibration nuclides

calibrationNuclides			
Field	Type	Length	Flags
idSample	int	4	PIFN
nuclideId	varchar	10	PIF1N
idCalibrationLibrary	int	4	F1
activity	double	8	
uncActivity	double	8	
assayTime	datetime	8	
comments	text	0	
(idSample) → samples(idSample)			
(nuclideId ,idCalibrationLibrary) → calibrationLibraries(nuclideId,idCalibrationLibrary)			

end of table.

This table contains information of the activity content of each nuclide of the calibration sample.

idSample	Identification number of the calibration sample. See table 6.1 samples .
nuclideId	Name of the calibration nuclide. See table 4.3 calibrationLibraries .
idCalibrationLibrary	Identification number of the library of nuclear data for calibration nuclides. See table 4.3 calibrationLibraries .
activity	Certified activity of the nuclide in Becquerels [Bq].
uncActivity	One sigma absolute uncertainty of activity in Becquerels [Bq].
assayTime	The assay date and time of the nuclide, i.e. the date and time of the activity .
comments	Comments in English [en].

4.3 Calibration Libraries

Table 4.3: Calibration libraries

calibrationLibraries			
Field	Type	Length	Flags
<code>idCalibrationLibrary</code>	int	4	PN
<code>nuclideId</code>	varchar	10	PIN
<code>idCalLine</code>	int	4	PN
<code>calLineEnergy</code>	double	8	
<code>uncCalLineEnergy</code>	double	8	
<code>emissionProb</code>	double	8	
<code>uncEmissionProb</code>	double	8	
<code>halflife</code>	double	8	
<code>halflifeUnit</code>	varchar	8	
<code>uncHalflife</code>	double	8	
<code>calibrationLibraryId</code>	varchar	20	
<code>comments</code>	text	0	

end of table.

This table contains gamma line data for calibration samples. Even though these data are fundamental constants, and thus one library should suffice, the table includes a possibility for multiple libraries. This may be advantageous for different calibration purposes where the useful gamma lines may differ.

Due to limited number of calibration sources available in a laboratory this table should not become too big. This is further emphasized by the fact that, unlike identification libraries, a calibration library usually contains data of the prominent gamma lines only.

<code>idCalibrationLibrary</code>	Identification number of the library.
<code>nuclideId</code>	Name of the calibration nuclide. The adopted syntax for <code>nuclideId</code> is Ba-137m, where <code>m</code> , <code>n</code> , <code>o</code> , ... denote metastable states of increasing energy. If metastable state energies are not known <code>a</code> , <code>b</code> , <code>c</code> , ... may be used. Note that analysis software may use different naming conventions. It is up to the database administrator to ascertain that consistent syntax is used. <i>Linssi</i> does not include a comprehensive nuclide reference library.
<code>idCalLine</code>	Index of the gamma line of the calibration nuclide. Sorted in an ascending order of the line energy starting from 1.
<code>calLineEnergy</code>	Energy of the gamma line of the calibration nuclide in kiloelectronvolts [keV].
<code>uncCalLineEnergy</code>	One sigma absolute uncertainty of <code>calLineEnergy</code> in kiloelectronvolts [keV].
<code>emissionProb</code>	Absolute emission probability of the gamma line, i.e., number of gammas emitted per decay.
<code>uncEmissionProb</code>	Absolute one sigma uncertainty of <code>emissionProb</code> .
<code>halflife</code>	Half-life of the nuclide in units <code>halflifeUnit</code> .
<code>halflifeUnit</code>	Half-life unit. The supported units are year [y] or [a], day [d], hour [h], minute [m] or [min], and second [s].

<code>uncHalflife</code>	Absolute one sigma uncertainty of <code>halflife</code> in units <code>halflifeUnit</code> .
<code>calibrationLibraryId</code>	Name of the calibration library.
<code>comments</code>	Comments in English [en].

Chapter 5

CTBT Laboratory Samples

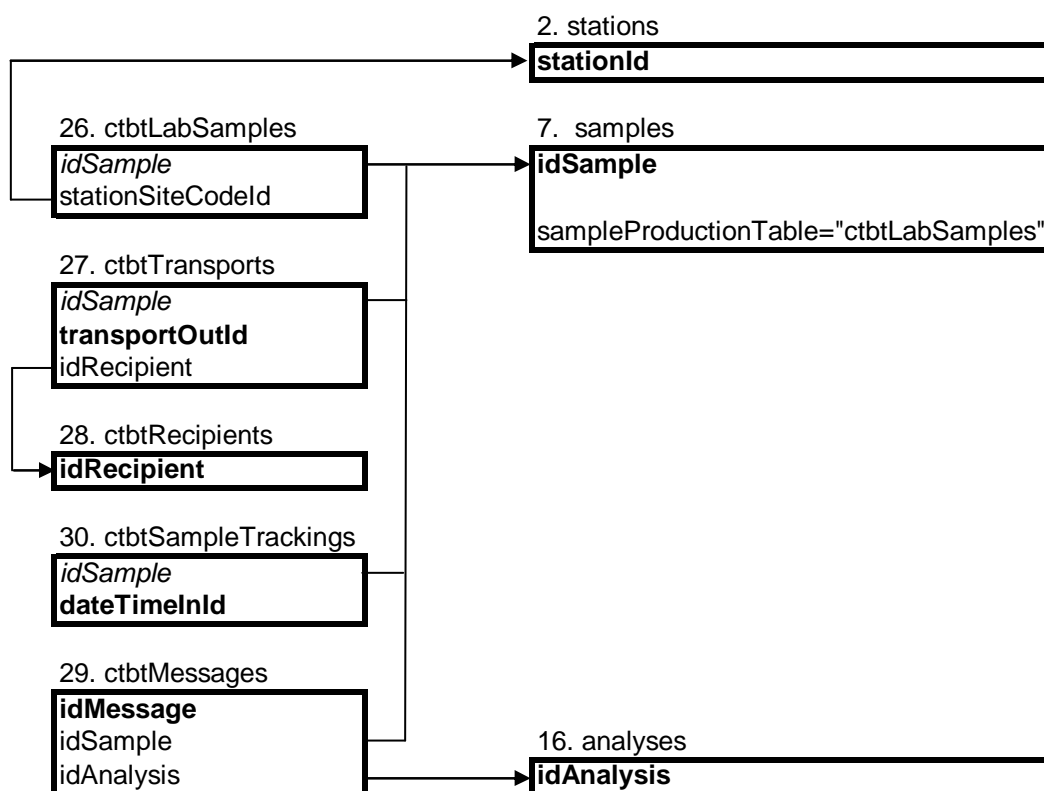


Figure 5.1: Tables of the CTBT sample production group (26, 27, 28, 29, and 30) and their connection to other *Linssi* tables (7 and 16).

These tables store information about CTBT laboratory samples sent to radionuclide laboratories within the framework of the Comprehensive Nuclear-Test-Ban Treaty. The tables are very specific and for the full details Ref. [6] should be consulted.

5.1 CTBT Laboratory Samples

Table 5.1: CTBT laboratory samples

ctbtLabSamples			
Field	Type	Length	Flags
idSample	int	4	PFN
siteCodeId	varchar	16	
stationSiteCodeId	varchar	16	F
priority	varchar	10	
sampleCategory	char	1	
collStart	datetime	8	
collEnd	datetime	8	
airVolumeTotal	double	8	
analysisPurpose	text	0	
testsAuthorized	text	0	
specialInstructions	text	0	
stationActCat	varchar	40	
stationSamDiam	double	8	
stationSamThick	double	8	
stationSamWidth	double	8	
stationSamMass	double	8	
stationContDens	double	8	
stationContThick	double	8	
stationContMat	varchar	80	
stationSamGeomDescr	varchar	80	
stationSplitMass	double	8	
stationSplitAirVolume	double	8	
stationSplitMethod	text	0	
idcNuclNotQuant	text	0	
idcNaturalNucl	text	0	
idcActProd	text	0	
idcFissProd	text	0	
idcEventActProdPresent	char	1	
idcEventDaysSinceLastAct	varchar	10	
idcEventOnlyOneFP	char	1	
idcEventDaysSinceLastFP	varchar	10	
idcEventMultFP	char	1	
idcEventDaysSinceLastMultFP	varchar	10	
idcEventCs137Present	char	1	
idcEventTimesCs137InMonth	varchar	10	
test	varchar	40	
comments	text	0	
(idSample) → samples(idSample)			
(stationSiteCodeId) → stations(stationId)			

end of table.

For a more precise definition of all the fields, except `idSample`, the user is asked to consult the report IDC3.4.1Rev6 [6] to which the descriptions below are referencing. The reference RLR/#xxx refers to the data block #xxx of the radionuclide laboratory report (RLR). RLR and the related data blocks are described in Chapter ‘Radionuclide Laboratory Reports’ of the report IDC3.4.1Rev6 [6].

<code>idSample</code>	Identification number of the sample measured. See table 6.1 samples.
<code>siteCodeId</code>	Site code of the laboratory selected for analysis. See RLR/#Header.
<code>stationSiteCodeId</code>	Site code of the sampling station.
<code>priority</code>	Priority level: Urgent or Routine. See RLR/#Header.
<code>sampleCategory</code>	Sample category: A - network quality control sample, B - sample from IMS network categorized by IDC as level 5, C - proficiency test sample, D - station backup sample (measured by a lab), or, E - other. See RLR/#Header.
<code>collStart</code>	Collection start date and time. See RLR/#Collection.
<code>collEnd</code>	Collection end date and time. See RLR/#Collection.
<code>airVolumeTotal</code>	Total air volume sampled in standard cubic meters [m ³]. See RLR/#Collection.
<code>analysisPurpose</code>	Purpose of the analysis. See RLR/#Objective.
<code>testsAuthorized</code>	Free text describing the tests authorized. See RLR/#Objective.
<code>specialInstructions</code>	Free text describing any special instructions. See RLR/#Objective.
<code>stationActCat</code>	Activity category of the sample according to international shipping regulations. See RLR/#StationSample.
<code>stationSamDiam</code>	Sample diameter or length in millimeters [mm]. See RLR/#StationSample.
<code>stationSamThick</code>	Sample thickness in millimeters [mm]. See RLR/#StationSample.
<code>stationSamWidth</code>	Sample width in millimeters [mm]. See RLR/#StationSample.
<code>stationSamMass</code>	Sample mass [g]. See RLR/#StationSample.
<code>stationContDens</code>	Sample container density [g/cm ³]. See RLR/#StationSample.
<code>stationContThick</code>	Sample container thickness in millimeters [mm]. See RLR/#StationSample.
<code>stationContMat</code>	Sample container material. See RLR/#StationSample.
<code>stationSamGeomDescr</code>	Sample geometry description. See RLR/#StationSample.
<code>stationSplitMass</code>	Mass of the split that gets to the lab. See RLR/#Split.
<code>stationSplitAirVolume</code>	Air volume of the split that gets to the lab. See RLR/#Split.
<code>stationSplitMethod</code>	Split method of the split that gets to the lab. See RLR/#Split.
<code>idcNuclNotQuant</code>	List of nuclides identified but not quantified by the IDC. See RLR/#g_IDCActivitySummary.
<code>idcNaturalNucl</code>	List of natural nuclides quantified by the IDC including their half-lives, concentrations [Bq/m ³] and absolute one sigma of uncertainties of concentrations. For the exact format see RLR/#g_IDCActivitySummary.

idcActProd	List of activation products quantified by the IDC including their half-lives, concentrations [Bq/m ³] and absolute one sigma of uncertainties of concentrations. For the exact format see RLR/#g_IDCActivitySummary.
idcFissProd	List of fission products quantified by the IDC including their half-lives, concentrations [Bq/m ³] and absolute one sigma of uncertainties of concentrations. For the exact format see RLR/#g_IDCActivitySummary.
idcEventActProdPresent	Activation products present in the sample [Y/N]. See RLR/#g_IDCEventScreeningFlags.
idcEventDaysSinceLastAct	Number of days since last activation product seen. See RLR/#g_IDCEventScreeningFlags.
idcEventOnlyOneFP	Only one fission product in the sample [Y/N] See RLR/#g_IDCEventScreeningFlags.
idcEventDaysSinceLastFP	Number of days since last fission product seen. See RLR/#g_IDCEventScreeningFlags.
idcEventMultFP	Two or more fission products in the sample [Y/N]. See RLR/#g_IDCEventScreeningFlags.
idcEventDaysSinceLastMult	Number of days since two or more fission products seen. See RLR/#g_IDCEventScreeningFlags.
idcEventCs137Present	Cs-137 present in the sample [Y/N]. See RLR/#g_IDCEventScreeningFlags.
idcEventTimesCs137InMonth	Number of times Cs-137 was seen in the last 30 days. See RLR/#g_IDCEventScreeningFlags.
test	Type of test performed. See RLR/#Test.
comments	Comments in English [en].

5.2 CTBT Transports

Table 5.2: CTBT transports

ctbtTransports			
Field	Type	Length	Flags
idSample	int	4	PFN
transportOutId	Boolean	1	PN
idRecipient	int	4	F
courierCompany	vchar	40	
dateTimeOfHandover	datetime	8	
eta	datetime	8	
airwayBill	vchar	40	
sealNumber	vchar	20	
comments	text	0	
(idSample) → samples(idSample)			
(idRecipient) → ctbtRecipients(idRecipient)			

end of table.

For a more precise definition of all the fields, except `idSample`, the user is asked to consult the report IDC3.4.1Rev6 [6] to which the descriptions below are referencing. The reference LABSDN/`#Transport` refers to the `#Transport` data block of the LABSDN message giving a notification that a sample has been sent to a laboratory. Same data blocks are also used in the TECSDN message giving a notification that a sample has been sent from the laboratory. LABSDN and TECSDN messages together with their data blocks are described in Chapter ‘Other Laboratory Messages’ of the report IDC3.4.1Rev6 [6].

<code>idSample</code>	Identification number of the sample measured. See table 6.1 <code>samples</code> .
<code>transportOutId</code>	True if this is a transport out of the laboratory, i.e., message TECSDN. See LABSDN/ <code>#Transport</code> .
<code>idRecipient</code>	Identification number of the recipient. See table 5.3 <code>ctbtRecipients</code> .
<code>courierCompany</code>	Name of the courier company. See LABSDN/ <code>#Transport</code> .
<code>dateTimeOfHandover</code>	Date and time of handover. See LABSDN/ <code>#Transport</code> .
<code>eta</code>	Estimated date and time of arrival. See LABSDN/ <code>#Transport</code> .
<code>airwayBill</code>	Airway bill number. See LABSDN/ <code>#Transport</code> .
<code>sealNumber</code>	Seal number. See LABSDN/ <code>#Transport</code> .
<code>comments</code>	Comments in English [en].

5.3 CTBT Recipients

Table 5.3: CTBT recipients

ctbtRecipients			
Field	Type	Length	Flags
<code>idRecipient</code>	int	4	PA
<code>pocName</code>	varchar	80	
<code>pocPhone</code>	varchar	40	
<code>pocOrg</code>	varchar	80	
<code>pocAddr</code>	varchar	80	
<code>pocEmail</code>	varchar	80	
<code>pocLocal</code>	Boolean	1	
<code>comments</code>	text	0	

end of table.

For a more precise definition of the fields the user is asked to consult the report IDC3.4.1Rev6 [6] to which the descriptions below are referencing. The reference LABSDN/#Recipient refers to the #Recipient data block of the LABSDN message giving a notification that a sample has been sent to a laboratory. Same data blocks are also used in the TECSDN message giving a notification that a sample has been sent from the laboratory. LABSDN and TECSDN messages together with their data blocks are described in Chapter ‘Other Laboratory Messages’ of the report IDC3.4.1Rev6 [6].

<code>idRecipient</code>	Identification number of the recipient.
<code>pocName</code>	Point of contact. See LABSDN/#Recipient.
<code>pocPhone</code>	Point of contact phone number. See LABSDN/#Recipient.
<code>pocOrg</code>	Point of contact organization. See LABSDN/#Recipient.
<code>pocAddr</code>	Point of contact address. See LABSDN/#Recipient.
<code>pocEmail</code>	Point of contact address. See LABSDN/#Recipient.
<code>pocLocal</code>	True if the point of contact is local.
<code>comments</code>	Comments in English [en].

5.4 CTBT Messages

Table 5.4: CTBT messages

ctbtMessages			
Field	Type	Length	Flags
idMessage	int	4	PA
messageId	varchar	20	N
senderSiteCode	varchar	16	N
idSample	int	4	F
idAnalysis	int	4	F
reportNumber	int	4	
messageType	varchar	40	
messageDataType	varchar	40	
reportType	varchar	3	
messageIdRef	varchar	20	
siteCodeMessageRef	varchar	16	
transmission	datetime	8	
receipt	datetime	8	
messageBody	longblob	0	
labDataVersion	varchar	20	
labDataVersionText	text	0	
receivedFrom	varchar	80	
sentTo	text	0	
ccTo	text	0	
authentication	Boolean	1	
authenticationPass	Boolean	1	
comments	text	0	
idcActSummary	text	0	
(idSample) → samples(idSample)			
(idAnalysis) → analyses(idAnalysis)			

end of table.

For a more precise definition of the fields the user is asked to consult the report IDC3.4.1Rev6 [6] to which the descriptions below are referencing. The reference RLR/#xxx refers to the data block #xxx of the radionuclide laboratory report (RLR). RLR and the related data blocks are described in Chapter ‘Radionuclide Laboratory Reports’ of the report IDC3.4.1Rev6 [6].

idMessage	<i>Linssi</i> identification number of the message.
messageId	Identification code of the message as represented in the message itself. See id_string of MSG_ID in Chapter ‘Message Structure’ of IDC3.4.1Rev6 [6].
senderSiteCode	Site code of facility, where the message is originated. See source in MSG_ID in Chapter ‘Message Structure’ of IDC3.4.1Rev6 [6].
idSample	Identification number of the sample measured. See table 6.1 samples.
idAnalysis	Identification number of the analysis. See table 8.1 analyses.

reportNumber	Number of the report. See RLR/#Header.
messageType	Message type, i.e., request, subscription, data, labdata, command_request, or command_response. See MSG_TYPE in Chapter ‘Message Structure’ of IDC3.4.1Rev6 [6].
messageDataType	Message data type. See DATA_TYPE in Chapter ‘Message Structure’ and the specific radionuclide data types in Chapter ‘Radionuclide Messages’ in IDC3.4.1Rev6 [6].
reportType	Message report type. Valid values are FIN and PREL. See RLR/#Header.
messageIdRef	A reference to the message for which this message is a response to, i.e., the ref_str of the REF_ID. See MSG_TYPE and REF_ID in Chapter ‘Message Structure’ of IDC3.4.1Rev6 [6].
siteCodeMessageRef	A reference to the message for which this message is a response to, i.e., ref_src in REF_ID in Chapter ‘Message Structure’ of IDC3.4.1Rev6 [6].
transmission	Date and time of the transmission of the message.
receipt	Date and time of the receipt of the message.
messageBody	The full text of the message.
labDataVersion	Version of message format used. See RLR/#LabDataVersion.
labDataVersionText	Notes regarding the message format. See RLR/#LabDataVersion.
receivedFrom	Email address of the sender (From:).
sentTo	Email address of the recipient (To:).
ccTo	Email address of the additional recipient (Cc:).
authentication	True if the message was authenticated.
authenticationPass	True if the authentication passed the checking.
comments	Comments in English [en].
idcActSummary	Summary of the IDC findings for this sample. See RLR/#Conclusions/~IDCSummary

5.5 CTBT Sample Tracking

Table 5.5: CTBT sample tracking

ctbtSampleTrackings			
Field	Type	Length	Flags
<code>idSample</code>	<code>int</code>	4	PFN
<code>dateTimeInId</code>	<code>datetime</code>	8	PN
<code>dateTimeOut</code>	<code>datetime</code>	8	
<code>location</code>	<code>varchar</code>	40	
<code>handledBy</code>	<code>varchar</code>	20	
<code>comments</code>	<code>text</code>	0	
(idSample) → samples(idSample)			

end of table.

This table is used to track the movements of the CTBT laboratory samples. The table itself is not specific to CTBT samples. The usage for other samples is pending on the overall design of the tracking system.

<code>idSample</code>	Identification number of the sample measured. See table 6.1 <code>samples</code> .
<code>dateTimeInId</code>	Date and time of the sample arrival to the location.
<code>dateTimeOut</code>	Date and time of the sample departure from the location.
<code>location</code>	Name of the location.
<code>handledBy</code>	Name of the person who transferred the sample to the location or is responsible for the sample at the location.
<code>comments</code>	Comments in English [en].

Chapter 6

Measurement

The core table of *Linssi* is `samples`, which describes the radioactive sample when it arrives to the measuring facility. This is **entry point 2** to *Linssi*. Every time an entry is created in *Linssi* tables it must have a corresponding entry in `samples`, or the entry must be simultaneously created. (The only exception is the station group of tables, which do not require a sample. See Ch. 2.)

The radioactivity contained in the sample is converted to the actual source (`sources`) which is measured using a setup comprising `measurementSetups`, `detectors`, `attenuators` and `shields` (Fig. 1.3). The data on the measurement and the resulting spectrum are stored in table `measurements`.

If the sample is split, or combined, its history can be reconstructed using the information in table `sampleSplitsCombines`.

6.1 Samples

Table 6.1: Sample description at arrival

samples			
Field	Type	Length	Flags
<code>idSample</code>	int	4	PA
<code>sampleId</code>	vchar	80	UN
<code>phdSampleName</code>	vchar	80	
<code>extSampleName</code>	vchar	80	
<code>splitSymbol</code>	vchar	8	
<code>split</code>	Boolean	1	
<code>combined</code>	Boolean	1	
<code>sampleType</code>	vchar	20	
<code>sampleProductionTable</code>	vchar	20	
<code>overallAct</code>	double	8	
<code>barcode</code>	text	0	
<code>sealNumberArrival</code>	vchar	20	
<code>sampleConditionArrival</code>	text	0	
<code>packConditionArrival</code>	text	0	
<code>sealConditionArrival</code>	text	0	
<code>sampleCondFlagArrival</code>	char	1	

continued on next page ...

... continued from previous page

Field	Type	Length	Flags
<code>sampleArrivalTime</code>	datetime	8	
<code>sampleReceivedBy</code>	varchar	20	
<code>dbEntryTime</code>	timestamp	4	
<code>comments</code>	text	0	

end of table.

This table describes a sample when it arrives for measurement. This is also the first entry point of the sample to the database, i.e., this record must always be the first record created for the sample. In many cases the only information at the time of creation may be the `sampleId`.

<code>idSample</code>	A unique auto incrementing identification number of the sample.
<code>sampleId</code>	A unique sample identifier. This character string follows the naming convention applied by the facility running the database.
<code>phdSampleName</code>	A sample reference identifier (SRID) string following the naming convention of the CTBTO/PTS [6]. Even though not required by the database this string is likely to be globally unique.
<code>extSampleName</code>	A sample identifier string following the naming convention of a customer for the analysis. Even though not required by the database this string is meant to be unique.
<code>splitSymbol</code>	A two part (Pp) split symbol of the sample. The symbol is used to identify the descendants of the original sample when it is split for multiple counting and analysis purposes. The first half, P, gives the split number, the second half, p, gives the total number of splits. The split identifier is thus 11 for all radionuclide samples before any splitting is performed. If, for example, a sample is split into three parts, the parts are assigned the following split identifiers: 13 for the first piece, 23 for the second piece, and 33 for the third piece. If all the sample splits are combined together, the split number P is set equal to p+1. Therefore, if the three sample splits from the previous example are combined together the split identifier reported is 43. This definition is a generalization of the one described in Ref. [6]. The split symbol is useful for tagging simple splits. For more complex cases of splitting and combining samples table 6.2 <code>sampleSplitsCombines</code> should be used. If both the table <code>sampleSplitsCombines</code> and <code>splitSymbol</code> are used, it is up to the user to make sure that they are consistent.
<code>split</code>	TRUE if the sample is formed by splitting other samples. For further information see table 6.2 <code>sampleSplitsCombines</code> .
<code>combined</code>	TRUE if the sample is formed by combining other samples. For further information see table 6.2 <code>sampleSplitsCombines</code> .
<code>sampleType</code>	Type of the sample. The following types are reserved: <code>air-filter</code> , <code>calibration</code> , <code>environmental</code> , <code>qcspectrum</code> , <code>blank</code> , <code>background</code> , <code>gassample</code> , <code>gasbackground</code> .

<code>sampleProductionTable</code>	Name of the table describing the creation of this sample. Currently tables <code>airFilterSamples</code> , <code>calibrationSamples</code> , and <code>ctbtLabSamples</code> have been defined.
<code>overallAct</code>	Approximate overall activity in becquerels [Bq]. Knowledge is useful when deciding the measurement setup and for radioprotection purposes.
<code>barcode</code>	Barcode on the sample.
<code>sealNumberArrival</code>	Sample package seal number.
<code>sampleConditionArrival</code>	Description of sample condition on arrival to measuring facility.
<code>packConditionArrival</code>	Description of sample package condition on arrival to measuring facility.
<code>sealConditionArrival</code>	Description of sample package seal condition on arrival to measuring facility.
<code>sampleCondFlagArrival</code>	Sample condition flag on arrival to measuring facility. The values of this flag are facility dependent.
<code>sampleArrivalTime</code>	Date and time of sample arrival to measuring facility.
<code>sampleReceivedBy</code>	Name of the person who received the sample.
<code>dbEntryTime</code>	Timestamp of entering this record to the database.
<code>comments</code>	Comments in English [en].

6.2 Splitting and Combining Samples

Table 6.2: Splitting and combining samples

sampleSplitsCombines			
Field	Type	Length	Flags
parentIdSample	int	4	PIFN
daughterIdSample	int	4	PIFN
activityBranching	double	8	
method	varchar	20	
comments	text	0	
(parentIdSample) → samples(idSample)			
(daughterIdSample) → samples(idSample)			

end of table.

A sample may be formed by splitting and combining other samples. In many cases it is enough to know simple sample properties, i.e., its history is irrelevant. However, if we want to know how the sample has been formed from other samples, it can be tracked with table `sampleSplitsCombines`.

This table is used together with table 6.1 `samples` where Boolean fields `split` and `combine` tell how the sample is formed. Note that those fields are useful, but not absolutely necessary, for tracking parents and daughters. For that purpose, a search through this table is the only thing needed.

<code>parentIdSample</code>	Identification number of the parent sample. See table 6.1 <code>samples</code> .
<code>daughterIdSample</code>	Identification number of the daughter sample. See table 6.1 <code>samples</code> .
<code>activityBranching</code>	Denotes the fraction of activity inherited from the parent to the daughter. If we sum <code>activityBranching</code> over all <code>daughterIdSample:s</code> while keeping <code>parentIdSample</code> fixed the result must be 1.0. <i>Note:</i> <code>activityBranching</code> should not be needed for the activity of the daughters. The metrics of each sample should be available in its production table.
<code>method</code>	Method of splitting or combining used to form the daughter from the parent(s).
<code>comments</code>	Comments in English [en].

6.3 Sources

Table 6.3: Source geometry and material description

sources			
Field	Type	Length	Flags
sourceId	varchar	40	PN
idSample	int	4	IF
sourceGeometry	varchar	20	
sourceThickness	double	8	
sourceHeightMar	double	8	
sourceWidth	double	8	
sourceLength	double	8	
sourceDiam1	double	8	
sourceDiam2	double	8	
sourceLayers	smallint	2	
sourceDensity	double	8	
sourceMass	double	8	
sourceMaterial	varchar	20	
contDens	double	8	
contThick	double	8	
contMaterial	varchar	40	
preparationMethod	text	0	
comments	text	0	

(idSample) → samples(idSample)

end of table.

This table describes the source geometry as it is when placed in the measuring position. If the source is processed from a sample, see *Case 2* below, it should contain all the radioactivity of the sample, except for decay and possible unavoidable losses in the preparation. If the sample has been deliberately split or combined, a new sample must be first formed and that sample should be the subject of source preparation.

This table aims to describe the geometry and material of the source, which includes the container, to the extent sufficient for Monte Carlo simulation of the measuring process when used together with the rest of the measurement setup.

sourceId	A unique name of the source.
idSample	Identifies the sample. See table 6.1 samples. We can identify two cases. <i>Case 1:</i> If the source is used to describe a standard measurement setup, i.e., its sourceId appears in table 6.7 measurementSetups, then idSample here should be set to null. In this case all the relevant samples are found from the calibration tables associated with the measurement setup in question.

Case 2: If the source is not describing a standard geometry, `idSample` must be given. In this case the analysis software is using the geometry information of the source to calculate a new, or to adjust an existing, calibration. See field `sourceId` in table 6.8 **measurements**.

<code>sourceGeometry</code>	Name of the source geometry. This name identifies the meaning of the source measures below.
<code>sourceThickness</code>	Thickness of the source in millimeters [mm].
<code>sourceHeightMar</code>	Height of the Marinelli beaker source in millimeters [mm].
<code>sourceWidth</code>	Width of the source in millimeters [mm].
<code>sourceLength</code>	Length of the source in millimeters [mm].
<code>sourceDiam1</code>	Outer diameter of the source in millimeters [mm].
<code>sourceDiam2</code>	Inner diameter of the source in millimeters [mm].
<code>sourceLayers</code>	Number of layers in the source.
<code>sourceDensity</code>	Density of the source in grams per cubic centimeter [g/cm ³].
<code>sourceMass</code>	Mass of the source in grams [g].
<code>sourceMaterial</code>	Name of the source material.
<code>contDens</code>	Density of the source container in grams per cubic centimeter [g/cm ³].
<code>contThick</code>	Thickness of the source container in millimeters [mm].
<code>contMaterial</code>	Name of the source container material.
<code>preparationMethod</code>	Describes how the source has been made from the sample.
<code>comments</code>	Comments in English [en].

6.4 Detectors

Table 6.4: Detectors

detectors			
Field	Type	Length	Flags
detectorId	varchar	40	PN
location	varchar	40	
detectorModel	varchar	40	
detectorType	varchar	20	
relEfficiency	double	8	
volume	double	8	
diameter	double	8	
thickness	double	8	
coreDiameter	double	8	
coreLength	double	8	
endcapToCrystal	double	8	
windowMaterial	varchar	20	
windowThickness	double	8	
deadLayerThickness	double	8	
biasVoltage	double	8	
polarity	varchar	10	
comments	text	0	

end of table.

This table describes germanium detectors used. This table aims to describe the detector to the extent sufficient for Monte Carlo simulation of the measuring process when used together with the rest of the measurement setup.

detectorId	A unique name of the detector.
location	Facility and location of the detector.
detectorModel	Detector manufacturer's model name and number for the detector.
detectorType	Type of the detector, e.g., p-type.
relEfficiency	Relative efficiency of the detector in percent [%].
volume	Volume of the detector in cubic millimeters [cm ³].
diameter	Detector diameter in millimeters [mm].
thickness	Detector thickness in millimeters [mm].
coreDiameter	Diameter of the detector core in millimeters [mm].
coreLength	Length of the detector core in millimeters [mm].
endcapToCrystal	Distance from the endcap to the crystal surface in millimeters [mm].
windowMaterial	Name of the detector window material.
windowThickness	Detector window thickness in millimeters [mm].
deadLayerThickness	Detector dead layer thickness in millimeters [mm].
biasVoltage	Bias voltage in volts [V].
polarity	Polarity of the voltage, positive or negative.
comments	Comments in English [en].

6.5 Attenuators

Table 6.5: Attenuators

attenuators			
Field	Type	Length	Flags
<code>attenuatorId</code>	<code>varchar</code>	20	PN
<code>comments</code>	<code>text</code>	0	

end of table.

This table describes attenuators used in measurements.

This table aims to describe the attenuator to the extent sufficient for Monte Carlo simulation of the measuring process when used together with the rest of the measurement setup.

`attenuatorId` A unique name of the attenuator.
`comments` Comments in English [en].

6.6 Shields

Table 6.6: Shields

shields			
Field	Type	Length	Flags
<code>shieldId</code>	<code>varchar</code>	20	PN
<code>comments</code>	<code>text</code>	0	

end of table.

This table describes shieldings used in measurements.

This table aims to describe the shielding to the extent sufficient for Monte Carlo simulation of the measuring process when used together with the rest of the measurement setup.

`shieldId` A unique name of the shield.
`comments` Comments in English [en].

6.7 Measurement Setups

Table 6.7: Measurement setups identify the components used in the setups and their positions

measurementSetups			
Field	Type	Length	Flags
measSetupId	varchar	40	PN
detectorId	varchar	40	F
sourceId	varchar	40	F
attenuatorId	varchar	40	F
shieldId	varchar	40	F
idCal	int	4	F
blankIdAnalysis	int	4	F
backgroundIdAnalysis	int	4	F
sampleDistX	double	8	
sampleDistY	double	8	
sampleDistZ	double	8	
attenDistX	double	8	
attenDistY	double	8	
attenDistZ	double	8	
shieldDistX	double	8	
shieldDistY	double	8	
shieldDistZ	double	8	
comments	text	0	

(detectorId) → detectors(detectorId)
(sourceId) → sources(sourceId)
(attenuatorId) → attenuators(attenuatorId)
(shieldId) → shields(shieldId)
(idCal) → calibrations(idCal)
(blankIdAnalysis) → analyses(idAnalysis)
(backgroundIdAnalysis) → analyses(idAnalysis)

end of table.

The Cartesian right hand coordinate system used here has its origin at the center of the detector's active front face window on its outer surface. The positive z-axis points outwards of the detector. If the z-axis is horizontal the y-axis is vertical and points upwards. Otherwise the y-axis is parallel to the plane of maximum possible symmetry and points outwards of the center of the detector system.

Sample, attenuator and shield coordinate axes are parallel to the coordinate axes defined here, but their origins are offset as described below.

The description of the measurement setup is meant to be accurate enough for Monte Carlo simulation of the measurements.

measSetupId	Unique name of the measurement setup.
detectorId	See table 6.4 detectors .
sourceId	See table 6.3 sources .

attenuatorId	See table 6.5 attenuators.
shieldId	See table 6.6 shields.
idCal	Identifies the correct calibration to be used with this setup. <i>Note:</i> each setup may have multiple calibrations but there is only one that is the best. See table 7.1 calibrations.
blankIdAnalysis	Identification number of the analysis of the blank. The analysis results of the blank have been used to set up the calibration for this measurement setup.
backgroundIdAnalysis	Identification number of the analysis of the background. The analysis results of the background have been used to set up the calibration for this measurement setup.
sampleDistX	x-component of the offset of the sample origin in millimeters [mm].
sampleDistY	y-component of the offset of the sample origin in millimeters [mm].
sampleDistZ	z-component of the offset of the sample origin in millimeters [mm].
attenDistX	x-component of the offset of the attenuator origin in millimeters [mm].
attenDistY	y-component of the offset of the attenuator origin in millimeters [mm].
attenDistZ	z-component of the offset of the attenuator origin in millimeters [mm].
shieldDistX	x-component of the offset of the shield origin in millimeters [mm].
shieldDistY	y-component of the offset of the shield origin in millimeters [mm].
shieldDistZ	z-component of the offset of the shield origin in millimeters [mm].
comments	Comments in English [en].

6.8 Measurements

Table 6.8: Measurement parameters and results

measurements			
Field	Type	Length	Flags
idMeas	int	4	PA
idSample	int	4	IFN
sourceId	varchar	40	F
measSetupId	varchar	40	F
blankIdMeas	int	4	FS
backgroundIdMeas	int	4	FS
measId	varchar	80	IUN
phdMeasName	varchar	80	
extMeasName	varchar	80	
sourcePreparedBy	varchar	20	
sourceReady	datetime	8	
detectorTemperature	double	8	
uncDetectorTemperature	double	8	
ambientTemperature	double	8	
ambientHumidity	double	8	
waitTime	double	8	
acqStart	datetime	8	
acqEnd	datetime	8	
acqRealTime	double	8	
acqLiveTime	double	8	
spectrumType	varchar	8	
firstChannel	int	4	
firstValidChannel	int	4	
lastValidChannel	int	4	
lastChannel	int	4	
spectrum	longblob	0	
spectrumSent	datetime	8	
comments	text	0	
(idSample) → samples(idSample) (sourceId) → sources(sourceId) (measSetupId) → measurementSetups(measSetupId) (blankIdMeas) → measurements(idMeas) (backgroundIdMeas) → measurements(idMeas)			

end of table.

idMeas
idSample
sourceId

A unique measurement identifier.

See table 6.1 samples.

The identifier of the measured source. See table 6.3 sources. We can identify two cases.

Case 1: If we are using a standard measurement setup without changes in the source geometry, then `sourceId` here should be set to `null`. In this case the relevant geometry information is found using `measSetupId` below.

Case 2: If the source is not describing a standard geometry the geometry is defined by `sourceId`. The analysis software is able to calculate a new, or adjust an existing, calibration by comparing the geometry given here with the geometry of the standard setup defined by the source associated with the standard geometry `measurementSetup.sourceId`.

<code>measSetupId</code>	See table 6.7 <code>measurementSetups</code> .
<code>blankIdMeas</code>	A name of the blank measurement relevant for this sample measurement. This is a unique identifier of the type <code>idMeas</code> pointing to a record in this <code>measurements</code> table.
<code>backgroundIdMeas</code>	A name of the background measurement relevant for this sample measurement. This is a unique identifier of the type <code>idMeas</code> pointing to a record in this <code>measurements</code> table.
<code>measId</code>	A unique measurement name, i.e., this has one-to-one correspondence with the primary key <code>idMeas</code> above. This name is normally a result of the naming rules applied in the measurement facility.
<code>phdMeasName</code>	A name following the naming rules of the measurement identification (MID) of the CTBTO [6]. Even though not required by the database this name is globally unique across different FULL spectra. However, the MID for eventual PREL spectra is identical to that of the FULL spectrum.
<code>extMeasName</code>	A name of the measurement given or required by the customer. This name is normally a result of the naming rules used by the customer. This name is most probably unique but it is not required by the database.
<code>sourcePreparedBy</code>	Name of the person who prepared the source.
<code>sourceReady</code>	Date and time the source was ready for first measurement.
<code>detectorTemperature</code>	Detector temperature in degrees centigrade [°C].
<code>uncDetectorTemperature</code>	One sigma absolute uncertainty of the detector temperature in degrees centigrade [°C].
<code>ambientTemperature</code>	Ambient temperature in degrees centigrade [°C].
<code>ambientHumidity</code>	Ambient relative humidity in percent [%].
<code>waitTime</code>	Time between end of sampling (or end of irradiation, or equivalent) and start of acquisition in seconds [s].
<code>acqStart</code>	Date and time of the start of acquisition.
<code>acqEnd</code>	Date and time of the end of acquisition.
<code>acqRealTime</code>	Real time of acquisition in seconds [s].
<code>acqLiveTime</code>	Measurement system effective live time of acquisition in seconds [s].
<code>spectrumType</code>	Type of the spectrum. Currently two types have been reserved: FULL denoting the normal full time spectrum, and PREL denoting a time slice of the full spectrum [6].
<code>firstChannel</code>	The channel number of the first number of counts in the <code>spectrum</code> .
<code>firstValidChannel</code>	The channel number of the first valid number of counts in the <code>spectrum</code> . This could be the first channel above the lower level discriminator, for example.

<code>lastValidChannel</code>	The channel number of the last valid number of counts in the <code>spectrum</code> . This could be the last channel below the upper level discriminator, for example.
<code>lastChannel</code>	The channel number of the last number of counts in the <code>spectrum</code> .
<code>spectrum</code>	The measured spectrum. Only the counts (y-values) are stored, i.e., no channel numbers (x-values). The format is one channel per line.
<code>spectrumSent</code>	Date and time when the spectrum was sent to analysis. If the analysis is carried out in the measuring facility, this is most probably the time the spectrum was stored to the database.
<code>comments</code>	Comments in English [en].

Chapter 7

Calibration Data and Functions

7.1 Calibrations

Table 7.1: Calibrations

Field	calibrations		
	Type	Length	Flags
idCal	int	4	PIN
calTypeId	varchar	20	PIN
measSetupId	varchar	40	F
inputIdCal	int	4	FS
changed	Boolean	1	
class	varchar	20	
description	varchar	40	
creationTime	datetime	8	N
calInfo	text	0	
calCertificate	text	0	
function	smallint	2	
functionDef	text	0	
startOfRange	double	8	
endOfRange	double	8	
par1	double	8	
par2	double	8	
par3	double	8	
par4	double	8	
par5	double	8	
par6	double	8	
par7	double	8	
par8	double	8	
par9	double	8	
par10	double	8	
uncPar1	double	8	
uncPar2	double	8	
uncPar3	double	8	

continued on next page ...

... continued from previous page

Field	Type	Length	Flags
uncPar4	double	8	
uncPar5	double	8	
uncPar6	double	8	
uncPar7	double	8	
uncPar8	double	8	
uncPar9	double	8	
uncPar10	double	8	
comments	text	0	
(measSetupId) → measurementSetups(measSetupId)			
(inputIdCal) → calibrations(idCal)			

end of table.

This table gives the calibration description and the fitted calibration function. Calibration data points can be found in table 7.2 `calPoints`. Note that in a single analysis multiple different calibrations (energy, efficiency, width, tail etc.) are needed. All these calibrations are given in this table and separated from each other by `calTypeId`. Calibrations with a same `idCal` in this table form a complete calibration used in analyses.

<code>idCal</code>	Identification number of the full calibration. The full calibration consists of multiple calibrations identified by <code>idCal.calTypeId</code> .
<code>calTypeId</code>	Name of the type of calibration. See Sec. 7.4 for discussion of the reserved <code>calTypeId</code> 's.
<code>measSetupId</code>	Identifier of the measurement setup for which this calibration has been performed. Even though there may be several calibrations for one measurement setup, only one can be the best.
<code>inputIdCal</code>	Pointer to the calibration that is the basis for this calibration. Using this pointer the calibration chain can be tracked to the original calibration. This key is a self referencing foreign key, i.e., it points to another record in this table.
<code>changed</code>	This field is true if the calibration <code>idCal.calTypeId</code> has been changed when compared with the calibration <code>inputIdCal.calTypeId</code> .
<code>class</code>	Additional information on the calibration chain. The following values have been defined: PHD: External calibration information received together with the spectrum. EXTERNAL: External calibration information of unspecified origin. SETUP: This is, or was, the default calibration for measurement setup <code>measSetupId</code> . <code>class</code> must have the value <code>SETUP</code> , if a <code>measurementSetups.idCal</code> refers to this calibration. <i>Note</i> : old default calibrations may retain this value even if there is no measurement setup referring to this calibration any more. SOH: A State Of Health calibration update generated by the spectrum analysis software. FIT: A (new) fit to the calibration points by the spectrum analysis software.

SOURCE: A corrected calibration due to source geometry. That is, the spectrum analysis software has made a correction to the calibration based on the differences in the measuring geometry between the source defined in the measurement setup and that actually measured. The correction may include source self-attenuation correction, source volume correction, source distance correction etc. For the corrections actually performed the software documentation must be consulted.

description	An identifier given by the analyst to the calibration (possibly unique, not required).
creationTime	Date and time of creation of the calibration. <i>Note:</i> A valid date and time must be given. If old calibrations are searched for, this field is needed as an identifier.
calInfo	Free form description of the calibration by the analyst.
calCertificate	Calibration certificate block as described in Ref. [6]. This is an alternative certificate of the calibration. The preferred method is described in Ch. 4. If both methods are used for the same calibration, they must be consistent.
function	Function identifier. See Ch. 7.5.
functionDef	Function definition. See Ch. 7.5.
startOfRange	Start of the validity range of the calibration function in channels [ch], or in kiloelectronvolts [keV] for efficiency calibrations.
endOfRange	End of the validity range of the calibration function in channels [ch], or in kiloelectronvolts [keV] for efficiency calibrations.
par1	Value of the 1 st parameter of the calibration function.
par2	Value of the 2 nd parameter of the calibration function.
par3	Value of the 3 rd parameter of the calibration function.
par4	Value of the 4 th parameter of the calibration function.
par5	Value of the 5 th parameter of the calibration function.
par6	Value of the 6 th parameter of the calibration function.
par7	Value of the 7 th parameter of the calibration function.
par8	Value of the 8 th parameter of the calibration function.
par9	Value of the 9 th parameter of the calibration function.
par10	Value of the 10 th parameter of the calibration function.
uncPar1	Uncertainty of the 1 st parameter of the calibration function.
uncPar2	Uncertainty of the 2 nd parameter of the calibration function.
uncPar3	Uncertainty of the 3 rd parameter of the calibration function.
uncPar4	Uncertainty of the 4 th parameter of the calibration function.
uncPar5	Uncertainty of the 5 th parameter of the calibration function.
uncPar6	Uncertainty of the 6 th parameter of the calibration function.
uncPar7	Uncertainty of the 7 th parameter of the calibration function.
uncPar8	Uncertainty of the 8 th parameter of the calibration function.
uncPar9	Uncertainty of the 9 th parameter of the calibration function.
uncPar10	Uncertainty of the 10 th parameter of the calibration function.
comments	Comments in English [en].

7.2 Calibration Points

Table 7.2: Calibration points

calPoints			
Field	Type	Length	Flags
idCalPoint	int	4	PN
idCal	int	4	PIF1N
calTypeId	varchar	20	PIF1N
idAnalysis	int	4	F2
idPeak	int	4	IF2
xValue	double	8	
yValue	double	8	
uncYValue	double	8	
comments	text	0	

(idCal ,calTypeId) → calibrations(idCal,calTypeId)
(idAnalysis ,idPeak) → peaks(idAnalysis,idPeak)

end of table.

This table contains calibration point values and their uncertainties given as triplets (`xValue`, `yValue`, `uncYValue`) sorted in an ascending order with respect to `xValue`.

It is not mandatory to have all the calibrations associated with spectrum analysis results. For those calibrations `idAnalysis = null`. This might be the situation in the case of off-line calibrations. It is, however, recommended that the user assigns an entry in the `analyses` table even for these cases. The entry and the corresponding `peaks` table provide ample space for justifying the calibration.

<code>idCalPoint</code>	Index of the calibration point. For each <code>idcal.calTypeId</code> the numbering starts from 1.
<code>idCal</code>	Identification number of the full calibration. The full calibration consists of multiple calibrations identified by <code>idCal.calTypeId</code> pairs. See table 7.1 <code>calibrations</code> .
<code>calTypeId</code>	Name of the type of calibration. See Ch. 7.4 for discussion of the reserved <code>calTypeId</code> 's.
<code>idAnalysis</code>	Identification number of the analysis used for this calibration point. See table 8.1 <code>analyses</code> .
<code>idPeak</code>	Identification number of spectrum peak for this calibration point. See table 8.2 <code>peaks</code> .
<code>xValue</code>	Channel, or energy [keV] for efficiency calibration, of the calibration point.
<code>yValue</code>	Value of the calibrated parameter at <code>xValue</code> .
<code>uncYValue</code>	The one sigma absolute uncertainty of the <code>yValue</code> .
<code>comments</code>	Comments in English [en].

7.3 Calibration Preferences

Table 7.3: Calibration preferences

calPreferences			
Field	Type	Length	Flags
<code>idCal</code>	int	4	PIFN
<code>idAnalysis</code>	int	4	PIFN
<code>usedInAnalysis</code>	Boolean	1	N
(idCal) → calibrations(idCal)			
(idAnalysis) → analyses(idAnalysis)			

end of table.

This table provides many-to-many relationships between analysis results and calibrations. A single full calibration, `idCal`, may, of course, be used in many analyses, `idAnalysis`. It is also possible that a single analysis takes advantage of many calibrations. For illustration, the analysis software may receive a calibration with the spectrum, `class = PHD`, but the analyst selects to use the default calibration for the measurement setup in question, `class = SETUP`. Finally, the software finetunes the calibration for source geometry differences and uses the adjusted calibration, `class = SOURCE`, to obtain the final results. Note that only the last calibration is used to calculate the final results, i.e., `usedInAnalysis` will be TRUE for that calibration only. However, after the analysis, at the latest, all the three above mentioned calibrations are stored in *Linssi*.

For information about the `class` field see table 7.1 calibrations.

<code>idCal</code>	Identification number of the full calibration. The full calibration consists of multiple calibrations identified by <code>idCal.calTypeId</code> pairs. See table 7.1 calibrations.
<code>idAnalysis</code>	Identification number of the analysis, which has used, or has considered to use this calibration. See table 8.1 analyses.
<code>usedInAnalysis</code>	TRUE if this is the calibration on which the analysis results are based on.
<code>comments</code>	Comments in English [en].

7.4 Calibration Types

Each complete calibration, `idCal`, in *Linssi* consists of calibrations of more than one parameter of the measurement setup. Calibrations of these individual parameters are identified by the calibration type identifiers, `calTypeId`, in tables 7.1 `calibrations`. The exact meaning and support of each calibration type depends on the analysis software. The currently reserved calibration types are described below.

7.4.1 Peak Shape Calibrations

The current version of *Linssi* reserves the following `calTypeId`'s for Gaussian peak shape with extended exponential tails and a baseline step: `width`, `lowTail`, `highTail`, `lowTailExp`, `highTailExp` and `step`. These parameters are given as a function of channel in *Linssi*. For exact meaning of the parameters see UNISAMPO, SHAMAN and *Aatami* manuals [1, 2, 7].

7.4.2 Energy Calibration

For energy calibration of photopeak centroids `calTypeId energy` is reserved. Here energy is given in kiloelectronvolts [keV] as a function of channel.

7.4.3 Efficiency Calibration

For detector efficiency calibration two `calTypeId`'s are reserved: `efficiency` for the photopeak efficiency and `totalEfficiency` for the total efficiency of the detector. Both efficiencies are defined as a function of energy in kiloelectronvolts [keV]. The efficiencies are absolute and describe the full measurement setup.

7.5 Calibration Functions

In *Linssi* the measurement setup calibration functions are used in tables 7.1 `calibrations`. The functions are fitted to points given in table 7.2 `calPoints`. In the functions the dependent variable y is defined as a function of one independent variable x with varying number of fitted parameters a_i . x is given either in kiloelectronvolts [keV] for efficiency calibration or in channels for all other calibration functions. The fitted parameters are stored in `par1...par10` and their uncertainties in `uncPar1...uncPar10`.

There are two ways of defining the calibration functions. In the first method a set of predefined functions is used. That is indicated by a non zero value of the field `function`. The second method involves storing the function definition in Mathematical Markup Language MathML into `functionDef`. This is informed by setting `function` to zero.

7.5.1 Predefined Functions

Predefined functions are selected by the field `function` of table 7.1 `calibrations`. List of currently supported functions is shown below. In principle any function can be applied for any calibration type. In practice, however, the functions 5...9 are used only for efficiency calibration. The user is advised to consult his analysis software manuals for

physical interpretation and support for these functions. If new functions are needed, their definitions should be sent to *Linssi* administrative body who will reserve function values for them.

function=0, MathML function

The function definition is stored in Mathematical Markup Language, MathML, into `functionDef`. See Sec. 7.5.2 below.

function=1, Interpolation

No functional fitting is available. Interpolation between the calibration points is used. Depending on the software and the type of calibration different types of interpolation may be used, i.e., linear, quadratic, logarithmic etc.

function=2, Polynomial

$$y = \sum_{i=0}^n a_i x^i \quad (7.1)$$

function=3, Square root polynomial

$$y = \sum_{i=0}^n a_i x^{i/2} \quad (7.2)$$

function=4, Square root of polynomial

$$y = \sqrt{\sum_{i=0}^n a_i x^i} \quad (7.3)$$

function=5, Exponential rollover

$$y = a_0 \exp\left(\frac{-a_1}{x}\right)^{a_3} \left[1 - \exp\left(\frac{-a_2}{x}\right)^{a_4}\right] \quad (7.4)$$

function=6, Polynomial in $\ln y$ against $\ln x$

$$\ln y = \sum_{i=0}^n a_i [\ln x]^i \quad (7.5)$$

function=7, Polynomial in $\ln y$ against x

$$\ln y = \sum_{i=0}^n a_i x^{1-i} \quad (7.6)$$

function=8, Polynomial in $\ln y$ against $1/x$

$$\ln y = \sum_{i=0}^{n-1} a_i \left[\ln \frac{a_n}{x}\right]^i \quad (7.7)$$

function=9, Inverse exponential

$$y = \frac{1}{a_0 x^{-a_2} + a_1 x^{-a_3}} \quad (7.8)$$

function=99, Other

This function is not defined nor is it available in MathML notation. Text in `functionDef` may help in interpreting the type of the function, if any, in question.

7.5.2 MathML presentation of the functions

Mathematical Markup Language, MathML [8], is a markup language defined using Extensible Markup Language, XML [9]. If `function=0` a MathML presentation of the function definition is stored into `functionDef`. The first parameters of `par1...par10` until the first null value shall be used to evaluate the function value. The function itself shall be encapsulated as

```
<mathml xmlns:z="http://www.w3.org/1998/Math/MathML" >
...
</mathml>
```

where the `http`-reference is to the applicable schema of MathML itself and `z` is the namespace prefix adopted in *Linssi*. These are the only requirements set by *Linssi* database specifications.

7.5.3 MathML support in Shaman

Even though *Linssi* sets minimal requirements for MathML functions, it should be pointed out that analysis software may set more stringent limits on the MathML features it is able to take advantage of. As far as we know, currently the only analysis program able to use MathML input is SHAMAN [2]. It uses a MathML subset briefly discussed here.

In SHAMAN lambda calculus is used to evaluate the function values. The parameters of `par1...par10` until the first null value are associated with `bvar`'s of the lambda construct. The association is by order, i.e., the first parameter, `par1`, refers to the first `bvar`. The last `bvar` is the independent variable. A function definition starts with a list of `bvar`'s followed by a container tag `apply`, `ci`, `cn`, `or`, `piecewise`, which must evaluate to a single value.

The following MathML tags are supported in SHAMAN:

Qualifier: `bvar`

Arithmetic and functions: `plus`, `minus`, `times`, `divide`, `power`, `root` (degree tag not supported, hence only square root available), `exp`, `ln`, `sin`, `cos`, `tan`

Binary forms of logical operators: `eq`, `gt`, `lt`, `geq`, `leq`

Logical operators: `neq`, `not`, `and`, `or`

Containers: `lambda`, `ci`, `cn`, `apply`, `piecewise`, `piece`, `otherwise` (Type attributes of `ci` and `cn` are not supported, hence always treated as real.)

The application producing the calibration function in MathML lambda notation is responsible for the correctness and numerical robustness of the function. While Shaman does perform some rudimentary error checking on the function before use, the conversion of the MathML function may involve some reordering of the function terms, which while mathematically equivalent, may cause numerically unstable functions to break.

An example of a linear function, $a + bx$, is presented in the lambda calculus of MathML on the following page.

Assuming, for example, that `par1` and `par2` have values of 2.1 and 1.2, respectively, the function below evaluates to $2.1 + 1.2x$.

```
<mathml xmlns:z="http://www.w3.org/1998/Math/MathML" >
  <z:lambda>
    <z:bvar>
      <z:ci>a</z:ci>
    </z:bvar>
    <z:bvar>
      <z:ci>b</z:ci>
    </z:bvar>
    <z:bvar>
      <z:ci>x</z:ci>
    </z:bvar>
    <z:apply>
      <z:plus/>
      <z:ci>a</z:ci>
      <z:apply>
        <z:times/>
        <z:ci>b</z:ci>
        <z:ci>x</z:ci>
      </z:apply>
    </z:apply>
  </z:lambda>
</mathml>
```

See SHAMAN input parser manual for more details on how to write these lambda constructs [10].

Chapter 8

Analysis

The analysis group of tables is in the core of the *Linssi* database. *Linssi* is designed to be useful in a facility doing gamma-ray spectrum analysis. The sample collection and measurement can very well be performed elsewhere and only the analysis results produced in-house. This is the **entry point 3** to the database.

All the tables in this group are filled with gamma-ray peak analysis and identification software. The amount of information may vary depending on the software used. The number of fields in these tables is large and many programs are not able to provide enough information to fill them all. *Linssi* developers have been using UNISAMPO – SHAMAN and *Aatami* – SHAMAN chains of software to provide all the information in these tables.

Our philosophy has been to be able to store multiple analysis results, identified by `idAnalysis`, for any measurement without overwriting the earlier ones. This facilitates either a fully automatic pipeline in an once-through fashion, or an iterative approach where the earlier results are used as a starting point for further interactive or automatic processing. The best results are then identified in table 8.8 `finalResults`.

8.1 Analyses

Table 8.1: General data on peak analysis

analyses			
Field	Type	Length	Flags
<code>idAnalysis</code>	int	4	PA
<code>idMeas</code>	int	4	IFN
<code>idSample</code>	int	4	IFN
<code>blankIdAnalysis</code>	int	4	FS
<code>backgroundIdAnalysis</code>	int	4	FS
<code>inputIdAnalysis</code>	int	4	FS
<code>spectrumArrival</code>	datetime	8	
<code>analysisBegin</code>	datetime	8	
<code>analysisEnd</code>	datetime	8	
<code>inputParam</code>	text	0	
<code>interactiveLog</code>	text	0	

continued on next page ...

... continued from previous page

Field	Type	Length	Flags
type	varchar	20	
software	varchar	20	
swVersion	varchar	80	
analyst	varchar	20	
baseline	longblob	0	
strippedSpectrum	longblob	0	
peakSearchSignificance	longblob	0	
refTime1	datetime	8	
decayTime1	double	8	
refTime2	datetime	8	
decayTime2	double	8	
refConstants	text	0	
baselineMethod	text	0	
peaksMethod	text	0	
nuclideMethod	text	0	
uncCalcMethod	text	0	
lcMethod	text	0	
alpha	double	8	
beta	double	8	
searchStartChannel	mediumint	3	
searchEndChannel	mediumint	3	
searchThreshold	double	8	
numberOfPeaks	mediumint	3	
totalCounts	int	4	
comments	text	0	

(idMeas) → measurements(idMeas)
(idSample) → samples(idSample)
(blankIdAnalysis) → analyses(idAnalysis)
(backgroundIdAnalysis) → analyses(idAnalysis)
(inputIdAnalysis) → analyses(idAnalysis)

end of table.

This table contains general data about the analysis, its input and how it has been performed. Some general analysis results are also provided.

idAnalysis	Identification number of the analysis.
idMeas	Identification number of the measurement, i.e., spectrum analyzed. See table 6.8 measurements.
idSample	Identification number of the sample measured. See table 6.1 samples.
blankIdAnalysis	Identification number of the analysis of the blank. The analysis results of the blank are used in this analysis for blank subtraction.
backgroundIdAnalysis	Identification number of the analysis of the background. The analysis results of the background are used in this analysis for background subtraction.

<code>inputIdAnalysis</code>	Identification number of the analysis used as input for this analysis. This analysis may be a continuation of the <code>inputIdAnalysis</code> or <code>inputIdAnalysis</code> is an analysis of a similar sample and thus a good starting point for this analysis
<code>spectrumArrival</code>	Date and time when the spectrum arrived for analysis.
<code>analysisBegin</code>	Date and time when the spectrum analysis started.
<code>analysisEnd</code>	Date and time when the spectrum analysis ended.
<code>inputParam</code>	The most essential input parameters used for the analysis.
<code>interactiveLog</code>	Log storing the interactive commands for the analysis given by the analyst.
<code>type</code>	Type of the analysis. This could be interactive, batch, pipeline, for example.
<code>software</code>	Name of the software used in analysis.
<code>swVersion</code>	Version of the software used in analysis.
<code>analyst</code>	Name of the analyst.
<code>baseline</code>	Channel by channel spectrum baseline used or generated in the analysis. The channel range is identical to <code>measurements.spectrum</code> but the values are floating point values.
<code>strippedSpectrum</code>	Channel by channel stripped spectrum used or generated in the analysis, i.e., the spectrum minus the peak functions. The channel range and format are identical to <code>measurements.spectrum</code> but the values are floating point values.
<code>peakSearchSignificance</code>	Channel by channel peak search significance, i.e., significance values given by the peak search algorithm, very often a digital filter. The peak search significance, L_s , usually aims to be directly proportional to the decision limit L_c , i.e., $L_s = C \times L_c$ but the constant C is dependent on the method, and not always constant, either. For the exact definition the software manual should be consulted. <i>Note:</i> usually the maxima of L_s do not fall on integer channels. The channel range and format are identical to <code>measurements.spectrum</code> but the values are floating point values.
<code>refTime1</code>	First reference date and time which the activities may be corrected to.
<code>decayTime1</code>	Decay time in seconds [s] from start of spectrum acquisition to <code>refTime1</code> .
<code>refTime2</code>	Second reference date and time which the activities may be corrected to.
<code>decayTime2</code>	Decay time in seconds [s] from end of activity collection to <code>refTime2</code> .
<code>refConstants</code>	References to the physical constants used. (e.g., ENSDF version, year, etc.)
<code>baselineMethod</code>	Spectrum baseline calculation method.
<code>peaksMethod</code>	Spectrum peak analysis method.
<code>nuclideMethod</code>	Nuclide identification and activity calculation method.
<code>uncCalcMethod</code>	Method of calculating uncertainties.
<code>lcMethod</code>	Method of calculating the decision level L_c .

<code>alpha</code>	<code>alpha</code> defines the confidence, $1 - \alpha$, on the <i>a posteriori</i> decision for accepting a gamma peak. Value of α is used to reject against <i>error of the first kind</i> .
<code>beta</code>	<code>beta</code> defines the confidence, $1 - \beta$, on the <i>a priori</i> detection of a gamma line. Value of β is used to reject against <i>error of the second kind</i> .
<code>searchStartChannel</code>	Peak search starts from this channel of the spectrum.
<code>searchEndChannel</code>	Peak search ends to this channel of the spectrum.
<code>searchThreshold</code>	Value of the search threshold used given in the units of L_s . See <code>peakSearchSignificance</code> above. The value is dependent on the search method used.
<code>numberOfPeaks</code>	Number of peaks in the spectrum.
<code>totalCounts</code>	Total number of counts in the spectrum.
<code>comments</code>	Comments in English [en].

8.2 Peaks

Table 8.2: Peak analysis results

peaks			
Field	Type	Length	Flags
idPeak	int	4	PN
idAnalysis	int	4	PIFN
idMeas	int	4	IFN
idSample	int	4	IFN
centroidChannel	double	8	
uncCentroidChannel	double	8	
energy	double	8	
uncEnergy	double	8	
area	double	8	
uncArea	double	8	
height	double	8	
width	double	8	
fwhm	double	8	
fwtm	double	8	
lowTail	double	8	
highTail	double	8	
lowTailExp	double	8	
highTailExp	double	8	
step	double	8	
netCountRate	double	8	
uncNetCountRate	double	8	
efficiency	double	8	
uncEfficiency	double	8	
searchSignificance	double	8	
significance	double	8	
significanceFlag	char	1	
decisionLimit	double	8	
detectionLimit	double	8	
outOfRange	Boolean	1	
peakOrigin	varchar	20	
ROIstart	int	4	
ROIend	int	4	
ROIarea	double	8	
ROIindex	mediumint	3	
baselineArea	double	8	
baselinePerChannel	double	8	
uncBaselinePerChannel	double	8	
baselineStart	mediumint	3	
baselineEnd	mediumint	3	
baselineParam1	double	8	

continued on next page ...

... continued from previous page

Field	Type	Length	Flags
baselineParam2	double	8	
baselineParam3	double	8	
baselineParam4	double	8	
emissionRate	double	8	
uncEmissionRate	double	8	
backgroundCps	double	8	
uncBackgroundCps	double	8	
backgroundType	varchar	20	
blankCps	double	8	
uncBlankCps	double	8	
blankType	varchar	20	
comments	text	0	
(idAnalysis) → analyses(idAnalysis)			
(idMeas) → measurements(idMeas)			
(idSample) → samples(idSample)			

end of table.

This table contains analysis results of spectral peaks. One record is used for each peak analyzed. In addition to the normal software dependence of all the results, it should be noted that many peak parameters may either be taken directly from the calibrations or calculated from spectral data.

idPeak	Peak index, i.e., the number of the peak in the analysis <code>idAnalysis</code> . The list is sorted in an ascending order according to peak channel and starting from 1.
idAnalysis	Identification number of the analysis. See table 8.1 <code>analyses</code>
idMeas	Identification number of the measurement. See table 6.8 <code>measurements</code> .
idSample	Identification number of the sample measured. See table 6.1 <code>samples</code> .
centroidChannel	Channel of the peak centroid [ch].
uncCentroidChannel	One sigma absolute uncertainty of the centroid channel in channels [ch].
energy	Peak energy in kiloelectronvolts [keV].
uncEnergy	One sigma absolute uncertainty of peak energy in kiloelectronvolts [keV].
area	Peak area in counts. <i>Note</i> : baseline area has been subtracted but the contributions from the blank and background not.
uncArea	One sigma absolute uncertainty of <code>area</code> in counts.
height	Peak height in counts.
width	One sigma peak width in channels [ch].
fwhm	Full width at half maximum of the peak in channels [ch].
fwtm	Full width at tenth maximum of the peak in channels [ch].
lowTail	Distance of the starting point of the low tail from peak centroid in channels [ch].

<code>highTail</code>	Distance of the starting point of the high tail from peak centroid in channels [ch].
<code>lowTailExp</code>	Value of the exponent used for the low tail definition.
<code>highTailExp</code>	Value of the exponent used for the high tail definition.
<code>step</code>	Relative height of the step under the peak, i.e., step-height-to-peak-area-ratio.
<code>netCountRate</code>	Net count rate at the peak, i.e., peak-area-to-live-time-ratio in counts per second [1/s]. The contribution from blank and background have been subtracted.
<code>uncNetCountRate</code>	One sigma absolute uncertainty of the net count rate in counts per second [1/s].
<code>efficiency</code>	Absolute efficiency of the measuring system at the peak energy.
<code>uncEfficiency</code>	One sigma absolute uncertainty of the absolute efficiency.
<code>searchSignificance</code>	Peak significance L_s as defined by peak search algorithm. L_s has a local maximum at <code>centroidChannel</code> . See <code>analyses.peakSearchSignificance</code> .
<code>significance</code>	Peak significance in multiples of the decision limit [L_c].
<code>significanceFlag</code>	Peak significance flags. They may correspond, for example, to high, medium, low, insignificant etc. The levels are decided by the software.
<code>decisionLimit</code>	Decision limit L_c in counts for this peak (in this spectrum).
<code>detectionLimit</code>	Detection limit L_d in counts for peak at this position (assuming the baseline of this spectrum).
<code>outOfRange</code>	TRUE if the centroid of this peak is outside of the valid channel range.
<code>peakOrigin</code>	A set of peak flags giving the origin of the peak. For the reserved values see Tab. 8.3.
<code>ROIstart</code>	Starting channel of the Region-Of-Interest (ROI) in which this peak belongs to.
<code>ROIend</code>	Ending channel of the corresponding ROI.
<code>ROIarea</code>	Total number of counts in ROI.
<code>ROIindex</code>	Index identifying the ROI sorted in an ascending order according to <code>ROIstart</code> and starting from 1.
<code>baselineArea</code>	Area of the baseline under the peak in counts, i.e., integral of the baseline function over the interval around peak deemed appropriate by the software.
<code>baselinePerChannel</code>	Baseline area in counts divided by the number of channels included in its calculation.
<code>uncBaselinePerChannel</code>	One sigma absolute uncertainty of the <code>baselinePerChannel</code> in counts.
<code>baselineStart</code>	Validity of the baseline function starts at this channel.
<code>baselineEnd</code>	Validity of the baseline function ends at this channel.
<code>baselineParam1</code>	The first of the four parameters reserved to define the baseline function.
<code>baselineParam2</code>	The second of the four parameters reserved to define the baseline function.
<code>baselineParam3</code>	The third of the four parameters reserved to define the baseline function.

<code>baselineParam4</code>	The fourth of the four parameters reserved to define the baseline function.
<code>emissionRate</code>	Emission rate of the source in gammas per second, [1/s], at peak energy, i.e., peak area divided by the live measuring time and efficiency. The contribution from blank and background have been subtracted.
<code>uncEmissionRate</code>	One sigma absolute uncertainty of the emission rate.
<code>backgroundCps</code>	Contribution of the background to peak count rate in counts per second [1/s] as derived from <code>backgroundIdAnalysis</code> . See table 8.1 analyses. <i>Note:</i> these are peak counts not baseline counts.
<code>uncBackgroundCps</code>	One sigma absolute uncertainty of background count rate in counts per second [1/s].
<code>backgroundType</code>	Type of the background: <code>detector</code> , <code>detector+blank</code> , etc., where <code>detector</code> means background only and <code>detector+blank</code> gives the contribution from both the background and blank. If <code>detector+blank</code> is given, <code>blankCps</code> below must be zero.
<code>blankCps</code>	Contribution of the blank to peak count rate in counts per second [1/s] as derived from <code>blankIdAnalysis</code> . See table 8.1 analyses. <i>Note:</i> these are peak counts not baseline counts
<code>uncBlankCps</code>	One sigma absolute uncertainty of blank count rate in counts per second [1/s].
<code>blankType</code>	Type of the blank: <code>blank</code> , <code>blank+detector</code> , etc., where <code>blank</code> means blank only, i.e., the background contribution to blank has been subtracted or negligible, and <code>blank+detector</code> gives the contribution from both the background and blank. If <code>blank+detector</code> is given, <code>backgroundCps</code> above must be zero.
<code>comments</code>	Comments in English [en].

Note on peak origin flags

The syntax of Aatami [7] has been adopted for the peak origin flags. The available flags are shown in Tab. 8.3. The syntax requires that two flags from each of the four groups are given and the order shown in the table is followed. However, the second flag of the first group is always blank. Thus a peak found using Mariscotti peak search; fitted using Full fit keeping the centroid fixed; area also calculated with Full fit but, of course, keeping its value free; and taking the FWHM from peak shape calibration and thus keeping its value fixed in last fitting, would read: M F0F1C0

1. Peak finding source flags (one flag and a blank)	
Flag	Explanation
A	Artificial peak
B	Background peak
E	External
M	Mariscotti peak search
R	Residual peak search (stripped spectrum)
I	Manually inserted (in certain channel)
l	Manually inserted (in library position)
L	Library inserted (multiplet fitting)
N	Inserted by natural radionuclide model
S	Inserted by summation peak model
2. Centroid source flags (2 flags)	
Flag	Explanation
Q	Quick fit
F	Full fit
U	User defined
E	External software
R	Multiplet fitting
M	Mariscotti center of gravity
L	Converted library energy
T	Tight fitting
0	Fixed in last fitting
1	Free in last fitting
3. Net area source flags (2 flags)	
Flag	Explanation
Q	Quick fit
F	Full fit
U	User defined
E	External software
R	Multiplet fitting
C	Calculated from reference line
S	Summation
s	Quick summation
0	Fixed in last fitting
1	Free in last fitting
4. Full width at half maximum source flags (2 flags)	
Flag	Explanation
Q	Quick fit
F	Full fit
U	User defined
E	External software
C	Calibration
0	Fixed in last fitting
1	Free in last fitting

Table 8.3: Peak origin flags. See note on p. 68

8.3 Line Associations

Table 8.4: Line associations

lineAssociations			
Field	Type	Length	Flags
idAnalysis	int	4	PF1N
nuclideId	varchar	10	PN
idLine	smallint	2	PN
idPeak	int	4	IF1
idMeas	int	4	IFN
idSample	int	4	IFN
lineEnergy	double	8	
uncLineEnergy	double	8	
emissionProb	double	8	
uncEmissionProb	double	8	
CCfactor	double	8	
uncCCfactor	double	8	
lineSignificance	double	8	
explLevel	double	8	
lorentzGamma	double	8	
xray	Boolean	1	
background	Boolean	1	
annihilation	Boolean	1	
singleEscape	Boolean	1	
doubleEscape	Boolean	1	
xrayEscape	Boolean	1	
backscatter	Boolean	1	
coincSum	Boolean	1	
randomSum	Boolean	1	
neutronScatter	Boolean	1	
neutronCapture	Boolean	1	
userGiven	Boolean	1	
found	Boolean	1	
foundClose	Boolean	1	
thresholdLine	Boolean	1	
primaryLine	Boolean	1	
actMan	Boolean	1	
comments	text	0	

(idMeas) → measurements(idMeas)
 (idSample) → samples(idSample)
 (idAnalysis ,idPeak) → analyses(idAnalysis,idPeak)

end of table.

Line association results are stored in this table. For each gamma line of each identified nuclide in a single analysis there may be at maximum one spectrum peak associated with it. At maximum, since small library lines may not be visible in the spectrum. Put it the other

way round: for each spectrum peak there may be any number of library lines associated with it.

It is not necessary for each gamma line to be associated with a spectrum peak. These lines have been used by the analysis software to support in nuclide identification.

For unidentified spectrum peaks `idLine` is 0 and `nuclideId` is `NO_ID1`, `NO_ID2`, `NO_ID3`,...

<code>idAnalysis</code>	Identification number of the analysis. See table 8.1 analyses.
<code>nuclideId</code>	Name of a nuclide. If identified the LSQ activity of this nuclide is based on the line(s) in this table for which the <code>idPeak</code> is not null. See table 8.5 activities. For syntax see table 4.3 calibrationLibraries. If not identified, see <code>idLine</code> below.
<code>idLine</code>	Index of the nuclide line starting from 1. This is a unique internal number for nuclide line given by the analysis software. It is recommended that this is the number of the line in the nuclide reference library. However, for SHAMAN this is an internal index, since the properties of coincidence and escape lines are internally calculated in SHAMAN. If this index is 0 <code>idPeak</code> has not been identified, i.e., there are no lines associated with it. In that case the <code>nuclideId</code> is <code>NO_ID1</code> , <code>NO_ID2</code> , <code>NO_ID3</code> ,...
<code>idPeak</code>	Identification number of the peak if there is a peak associated with this gamma line. Otherwise <code>idPeak</code> is null. See table 8.2 peaks. This peak has been used in calculation of <code>actRawLSQ</code> . Depending on values of <code>actMan</code> and <code>primaryLine</code> this peak has been used also in calculation of <code>actRawMan</code> and <code>actRawPriLine</code> . See below.
<code>idMeas</code>	Identification number of the measurement. See table 6.8 measurements.
<code>idSample</code>	Identification number of the sample measured. See table 6.1 samples.
<code>lineEnergy</code>	Energy of the gamma line in kiloelectronvolts [keV]. <i>Note:</i> this is not the energy of the gamma peak in the spectrum.
<code>uncLineEnergy</code>	Absolute one sigma uncertainty of <code>lineEnergy</code> .
<code>emissionProb</code>	Emission probability of the gamma line (fraction, NOT in %).
<code>uncEmissionProb</code>	One sigma absolute uncertainty of the emission probability.
<code>CCfactor</code>	True coincidence correction factor of the gamma line.
<code>uncCCfactor</code>	One sigma absolute uncertainty of the true coincidence correction factor of the gamma line.
<code>lineSignificance</code>	The significance of the gamma line in units of the detection level L_d . This is the calculated line significance based on the activity of the identified nuclide. It gives a measure whether this gamma line should be visible in the spectrum as a gamma peak or not.
<code>explLevel</code>	The fraction of peak area explained by this gamma line.
<code>lorentzGamma</code>	Value of the Lorentz gamma width of this X-ray line in kiloelectronvolts [keV].
<code>xray</code>	TRUE if this line is an X-ray line.
<code>background</code>	TRUE if this line is a background line.
<code>annihilation</code>	TRUE if this line is an annihilation line.
<code>singleEscape</code>	TRUE if this line is a single escape line.
<code>doubleEscape</code>	TRUE if this line is a double escape line.
<code>xrayEscape</code>	TRUE if this line is an X-ray escape line.

<code>backscatter</code>	TRUE if this line is a backscatter line.
<code>coincSum</code>	TRUE if this line is a true coincidence sum line.
<code>randomSum</code>	TRUE if this line is a random coincidence sum line.
<code>neutronScatter</code>	TRUE if this line is due to neutron scatter in the detector.
<code>neutronCapture</code>	TRUE if this line is due to neutron capture in the detector.
<code>userGiven</code>	TRUE if this line is a user defined line.
<code>found</code>	TRUE if this line is found, i.e., associated with a peak in the spectrum. The LSQ calculation of nuclide activity is based on all nuclide lines for which <code>found</code> is TRUE. See table 8.5 activities.
<code>foundClose</code>	TRUE if this line is found close to a spectrum peak but not associated with it.
<code>thresholdLine</code>	TRUE if this line is the threshold line. The threshold line is the most significant line of a nuclide that has NOT been associated with a peak in the spectrum.
<code>primaryLine</code>	TRUE if this line is the primary line, i.e., the most significant line of the nuclide given the spectral baseline. The primary line activity <code>actRawPriLine</code> is based on the peak associated with this gamma line. See table 8.5 activities.
<code>actMan</code>	TRUE if this line has been used in the off-line, possibly manual, calculation of the nuclide activity <code>actRawMan</code> . In that case the line must have been associated with a peak, i.e., <code>idPeak</code> must not be null. See table 8.5 activities.
<code>comments</code>	Comments in English [en].

8.4 Nuclides and Their Activities

Table 8.5: Identified nuclides and their activities

activities			
Field	Type	Length	Flags
nuclideId	varchar	10	PIN
idAnalysis	int	4	PIFN
idMeas	int	4	IFN
idSample	int	4	IFN
confidence	double	8	
effHalflife	double	8	
effHalflifeUnit	varchar	8	
uncEffHalflife	double	8	
isBackground	Boolean	1	
actRawLSQ	double	8	
uncActRawLSQ	double	8	
interfCorrLSQ	Boolean	1	
actRawPriLine	double	8	
uncActRawPriLine	double	8	
interfCorrPriLine	Boolean	1	
actRawMan	double	8	
uncActRawMan	double	8	
actRawManMethod	text	0	
actSelect	varchar	10	
acqCorr	double	8	
uncAcqCorr	double	8	
decayCorr1	double	8	
uncDecayCorr1	double	8	
waitCorr	double	8	
uncWaitCorr	double	8	
irrCorr	double	8	
uncIrrCorr	double	8	
collCorr	double	8	
uncCollCorr	double	8	
decayCorr2	double	8	
uncDecayCorr2	double	8	
comments	text	0	
(idAnalysis) → analyses(idAnalysis)			
(idMeas) → measurements(idMeas)			
(idSample) → samples(idSample)			

end of table.

Identified nuclides and their raw activities are stored in this table. The table contains activities calculated using only the primary lines and the activities where all found peaks of the nuclides are used in the least squares (LSQ) sense. The interfering nuclides may have been resolved during the process. There is also a field for off-line calculated raw activity. If the contributions from blank and background are known (see table 8.2 peaks) their contribution

has been subtracted, i.e., peaked background subtraction (PBS) has been applied to all activities.

The decay correction factors, for evaluation of the nuclide activity at specific dates and times from the raw activities, can also be stored. If the decay chain is in equilibrium these factors have been calculated using the effective half-lives of the nuclides. In the case of a decay chain not in equilibrium the effective half-lives cannot be used. Even in this case the given correction factors can be used to obtain the decay corrected activities. The user should, however, consult the analysis software manual to find out whether the assumption of equilibrium has been applied or not. Since the correction factors are cumulative it would be tempting to calculate the total uncertainty from the uncertainties of the individual factors using the normal error propagation law of Gauss. However, since the decay corrections are strongly correlated due to identical half-lives, it must also be taken into account. The corrections and the associated times are illustrated in Fig. 8.1 below. Decay correction factors are multiplicative.

<code>nuclideId</code>	Name of the identified nuclide. For syntax see table 4.3 calibrationLibraries.
<code>idAnalysis</code>	Identification number of the analysis. See table 8.1 analyses.
<code>idMeas</code>	Identification number of the measurement. See table 6.8 measurements.
<code>idSample</code>	Identification number of the sample measured. See table 6.1 samples.
<code>confidence</code>	A figure of credit, p , describing the confidence on the presence of this nuclide in the sample, i.e., there is a probability of $1 - p$ that this nuclide is not present. <i>Note:</i> identification is a binary decision where valid calculations of the statistics are extremely difficult. See the manual of the software producing this number.
<code>effHalflife</code>	Effective half-life of the nuclide in units <code>effHalflifeUnit</code> .
<code>effHalflifeUnit</code>	Half-life unit. The supported units are year [y] or [a], day [d], hour [h], minute [m] or [min], and second [s].
<code>uncEffHalflife</code>	Absolute one sigma uncertainty of <code>effHalflife</code> in units <code>effHalflifeUnit</code> .
<code>isBackground</code>	True if background efficiency is used to calculate the activity. This is the activity due to surrounding walls etc., i.e., not the activity in the sample. Normally its absolute value is not known and only the shape of the background efficiency curve has been used to facilitate more reliable nuclide identification. This background should not be confused with the peaked background subtraction (PBS) due to background measurement. As a matter of fact, they are incompatible and should not be used together.
<code>actRawLSQ</code>	Raw activity in becquerels [Bq] based on all the peaks of this nuclide. Weighted least squares fitting (LSQ) has been used to obtain the effective peak area. Raw activity is the net peak area multiplied by <code>CCfactor</code> (see table 8.4 lineAssociations), and divided by efficiency, live measuring time and emission probability of the gamma line. The contribution from blank and background has been subtracted. See also note on specific activity on p. 79.
<code>interfCorrLSQ</code>	TRUE if the interference due to other identified nuclides has been accounted for in <code>actRawLSQ</code> .
<code>uncActRawLSQ</code>	One sigma absolute uncertainty in becquerels [Bq] of <code>actRawLSQ</code> .

<code>actRawPriLine</code>	Raw activity in becquerels [Bq] based on the primary line of this nuclide. Raw activity is the net peak area multiplied by <code>CCfactor</code> (see table 8.4 <code>lineAssociations</code>), and divided by efficiency, live measuring time and emission probability of the gamma line. The contribution from blank and background has been subtracted. See also note on specific activity on p. 79.
<code>interfCorrPriLine</code>	TRUE if the interference due to other identified nuclides has been accounted for in <code>actRawPriLine</code> .
<code>uncActRawPriLine</code>	One sigma absolute uncertainty in becquerels [Bq] of <code>actRawPriLine</code> .
<code>actRawMan</code>	Off-line, possibly manually, calculated raw activity in becquerels [Bq]. See also note on specific activity on p. 79.
<code>uncActRawMan</code>	One sigma absolute uncertainty in becquerels [Bq] of <code>actRawMan</code> .
<code>actRawManMethod</code>	Method used to calculate <code>actRawMan</code> .
<code>actSelect</code>	Identifies the activity used to calculate the final results. The possible values are LSQ: <code>actRawLSQ</code> has been used in table 8.8 <code>finalResults</code> . PRI: <code>actRawPriLine</code> has been used in table 8.8 <code>finalResults</code> . MAN: <code>actRawMan</code> has been used in table 8.8 <code>finalResults</code> . The decision which activity is to be used must be made by the analyst. That is, the analysis software must set this field to <code>null</code> . Which gamma lines have been used in the activity calculation can be found in table 8.4 <code>lineAssociations</code> .
<code>acqCorr</code>	Acquisition correction multiplier corrects for decay during spectrum measurement. See <code>acqRealTime</code> in table 8.1 <code>analyses</code> .
<code>uncAcqCorr</code>	One sigma absolute uncertainty of <code>acqCorr</code> .
<code>decayCorr1</code>	Decay correction multiplier #1. See <code>decayTime1</code> in table 8.1 <code>analyses</code> .
<code>uncDecayCorr1</code>	One sigma absolute uncertainty of <code>decayCorr1</code> .
<code>waitCorr</code>	Decay correction multiplier for wait time. See <code>waitTime</code> in table 6.8 <code>measurements</code> .
<code>uncWaitCorr</code>	One sigma absolute uncertainty of <code>waitCorr</code> .
<code>irrCorr</code>	This correction takes into account the activity production rate and decay during irradiation/collection. When applied to the activity at the end of collection it gives the saturation activity of the sample, i.e., the activity assuming infinite irradiation time with the average activity production rate. <code>irrCorr</code> depends on the time profile of the collection rate. If it is not known, a common assumption is a constant rate. See your software documentation for the actual method used.
<code>uncIrrCorr</code>	One sigma absolute uncertainty of <code>irrCorr</code> .
<code>collCorr</code>	This correction takes into account the activity production/collection rate and decay during collection. When applied to the activity at the end of collection it gives the total activity collected. <code>collCorr</code> depends on the time profile of the collection rate. If it is not known, a common assumption is a constant rate. See your software documentation for the actual method used.
<code>uncCollCorr</code>	One sigma absolute uncertainty of <code>collCorr</code> .

<code>decayCorr2</code>	Decay correction multiplier #2. See <code>decayTime2</code> in table 8.1 analyses.
<code>uncDecayCorr2</code>	One sigma absolute uncertainty of <code>decayCorr2</code> .
<code>comments</code>	Comments in English [en].

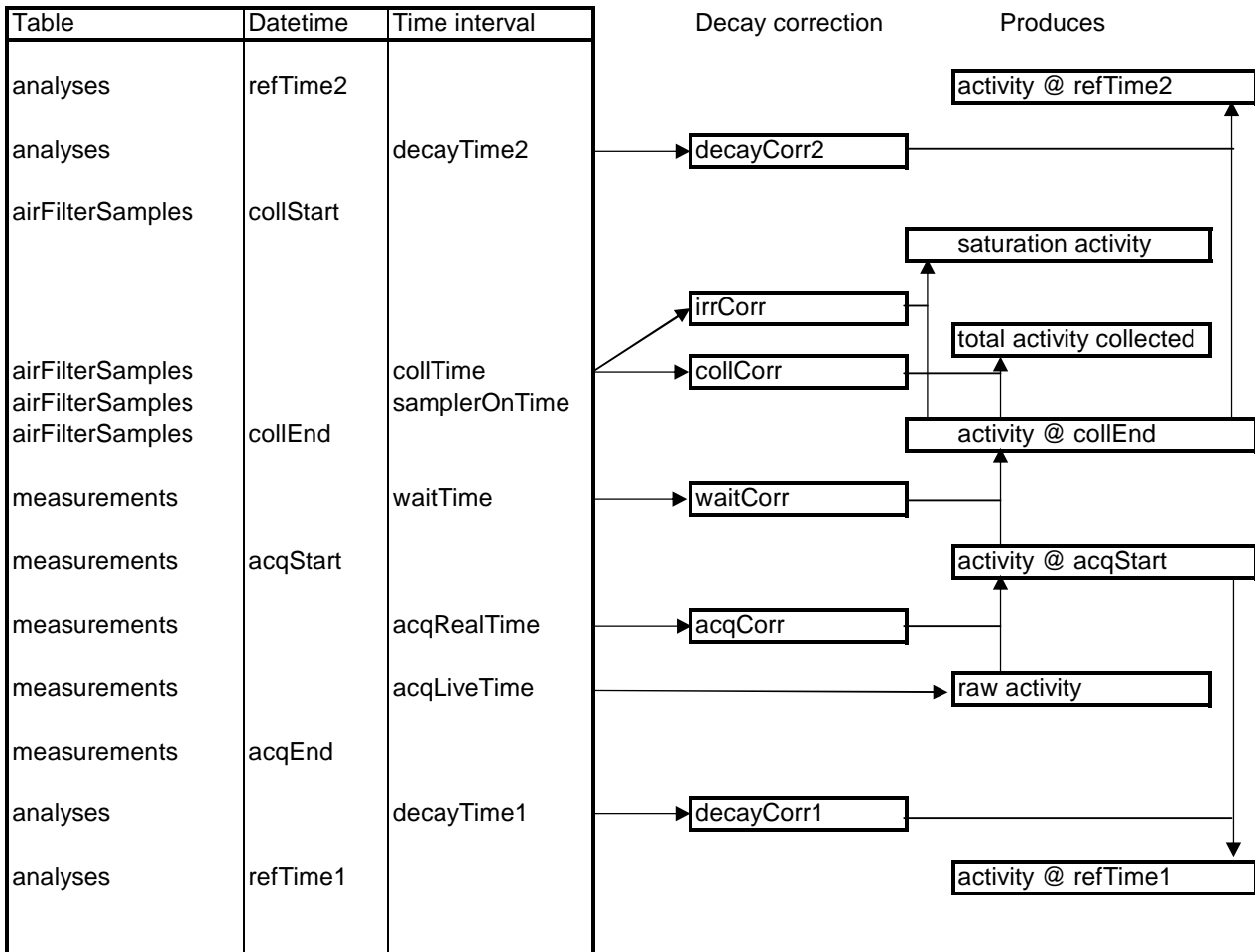


Figure 8.1: Decay corrections in *Linssi*. The time intervals used to calculate the various corrections are shown in the left hand part of the figure together with the times and dates from which they are calculated. It also includes the names of the table where the corresponding fields can be found. The right hand part of the figure shows the decay correction factors in table `activities` and how they are used to obtain the decay corrected activities at given dates and times. The activity at the point of an arrow is obtained by **multiplying** the activity at the tail of the arrow with the correction factor shown.

Note 1: Raw activity, saturation activity and total activity collected are not activities at any specific date and time.

Note 2: `collCorr` and `irrCorr` depend on the shape of the collection rate profile and generally cannot be calculated from `collTime` and `samplerOnTime` alone.

Note 3: Even though the collection refers to table `airFilterSamples` the same correction is also applicable to other sample production methods.

Note 4: `decayTime1` and `decayTime2` may be positive or negative. The other times, by definition, are positive.

Note 5: If the decay chains are in equilibrium the decay corrections can be easily obtained from the effective half-lives of the identified nuclides. In non-equilibrium chains the effective half-lives do not exist. Depending on the analysis software the correction factors may still be correct. Consult your software manual.

8.5 Activity Limits

Table 8.6: Activity limits and minimum detectable activities

activityLimits			
Field	Type	Length	Flags
nuclideId	varchar	10	PN
idAnalysis	int	4	PIFN
idMeas	int	4	IFN
idSample	int	4	IFN
energyPriLine	double	8	
area	double	8	
uncArea	double	8	
baselineArea	double	8	
uncBaselineArea	double	8	
decisionLimit	double	8	
detectionLimit	double	8	
mda	double	8	
mdc	double	8	
significance	double	8	
significanceFlag	char	1	
comments	text	0	
(idAnalysis) → analyses(idAnalysis)			
(idMeas) → measurements(idMeas)			
(idSample) → samples(idSample)			

end of table.

There are many methods to calculate minimum detectable activities (MDA) and the manual of the software performing the analysis should always be consulted for exact meaning of the fields below. It should be noted that the peak area may be based on the peak actually found in peak analysis or its calculation may have been forced by the MDA-algorithm itself. It is also possible that background peaks or interfering nuclides have been taken into account in the process of defining the peak and baseline areas.

nuclideId	Name of the nuclide for which this MDA is calculated. <i>Note:</i> this nuclide does not need to be a nuclide identified in the spectrum. For identified nuclides more data can be found in table 8.5 activities. For syntax see table 4.3 calibrationLibraries.
idAnalysis	Identification number of the analysis. See table 8.1 analyses.
idMeas	Identification number of the measurement. See table 6.8 measurements.
idSample	Identification number of the sample measured. See table 6.1 samples.

<code>energyPriLine</code>	Energy of the primary line of this nuclide in kiloelectronvolts [keV]. <i>Note:</i> the primary line is not necessary the one with the highest emission probability. The primary line is defined by the analysis software on the basis of maximum ‘visibility’ in this analysis. The MDA is calculated on the basis of this line.
<code>area</code>	Area of the peak at energy <code>energyPriLine</code> in counts.
<code>uncArea</code>	Absolute one sigma uncertainty of <code>area</code> in counts.
<code>baselineArea</code>	Area of the baseline at energy <code>energyPriLine</code> in counts used to calculate activity limits. <i>Note:</i> this is NOT the <code>baselineArea</code> of table 8.2 peaks.
<code>uncBaselineArea</code>	Absolute one sigma uncertainty of <code>baselineArea</code> in counts.
<code>decisionLimit</code>	Decision limit in counts. This corresponds the value of α in 8.1 analyses.
<code>detectionLimit</code>	Detection limit in counts. This corresponds the value of β in 8.1 analyses.
<code>mda</code>	Minimum detectable activity in becquerels [Bq] decay corrected to start of counting. See also the note on specific activity below.
<code>mdc</code>	Minimum detectable concentration. The unit may be Bq/m ³ , Bq/g, etc. depending on the application. It is based on the raw activity as defined in Fig. 8.1 and table 8.5 activities. See also the note on specific activity below.
<code>significance</code>	Peak area in the units of decision limit, i.e., <code>area</code> divided by <code>decisionLimit</code> .
<code>significanceFlag</code>	Flags indicating peak significance. The valid values are software dependent, in UNISAMPO 0 and 1.
<code>comments</code>	Comments in English [en].

Note on specific activity

The activities in table 8.5 activities are raw sample activities in Bq. There are, however, cases where sample activities cannot be defined. For example, an *in situ* measurement of the surface activity of soil from the distance of 1 m above the ground does not give the total surface activity of the globe in Bq. In cases like this, efficiency calibration must be performed to give directly the specific activity, in this case activity per area at the measuring position, the unit being Bq/m². Accordingly, the MDA-values in the above table refer to the corresponding specific activity. In this case, distinction between MDA and MDC cannot generally be made and they are thus equal.

What is the denominator in the unit of the specific activity depends on the specific sample production model and is given in the sample production groups of tables, Fig. 1.3. In the case where the MDA is real activity in Bq, i.e., not the specific activity, these tables also contain the numerical value used to divide the MDA to get the MDC.

8.6 Nuclide Ratios

Table 8.7: Activity ratios of relevant nuclide pairs

nuclideRatios			
Field	Type	Length	Flags
<code>idAnalysis</code>	int	4	PIF12N
<code>firstNuclideId</code>	varchar	10	PF1N
<code>secondNuclideId</code>	varchar	10	PF2N
<code>idMeas</code>	int	4	IFN
<code>idSample</code>	int	4	IFN
<code>firstIsDaughter</code>	Boolean	1	
<code>secondIsDaughter</code>	Boolean	1	
<code>firstHalflife</code>	double	8	
<code>uncFirstHalflife</code>	double	8	
<code>secondHalflife</code>	double	8	
<code>uncSecondHalflife</code>	double	8	
<code>halflifeUnit</code>	varchar	8	
<code>netBranching</code>	double	8	
<code>uncNetBranching</code>	double	8	
<code>refRatio</code>	double	8	
<code>uncRefRatio</code>	double	8	
<code>zeroRatio</code>	double	8	
<code>uncZeroRatio</code>	double	8	
<code>refTime</code>	datetime	8	
<code>zeroTime</code>	datetime	8	
<code>uncZeroTimeLow</code>	double	8	
<code>uncZeroTimeHigh</code>	double	8	
<code>comments</code>	text	0	

(`idMeas`) → `measurements(idMeas)`
 (`idSample`) → `samples(idSample)`
 (`idAnalysis` ,`firstNuclideId`) → `activities(idAnalysis,nuclideId)`
 (`idAnalysis` ,`secondNuclideId`) → `activities(idAnalysis,nuclideId)`

end of table.

This table contains the information of activity ratios of nuclide pairs that facilitates calculation of the birth time of activity, `zeroTime`. The nuclides may belong to the same decay chain or decay independently. If they decay independently, it is, of course, necessary to have *á priori* information on their relative yields, `zeroRatio`, at time zero.

<code>idAnalysis</code>	Identification number of the analysis. See table 8.1 analyses.
<code>firstNuclideId</code>	Name of the first nuclide. See table 8.5 activities.
<code>secondNuclideId</code>	Name of the second nuclide. See table 8.5 activities.
<code>idMeas</code>	Identification number of the measurement. See table 6.8 measurements.
<code>idSample</code>	Identification number of the sample measured. See table 6.1 samples.
<code>firstIsDaughter</code>	The first nuclide follows the second in the decay chain.

<code>secondIsDaughter</code>	The second nuclide follows the first in the decay chain. <i>Note 1:</i> Both <code>firstIsDaughter</code> and <code>secondIsDaughter</code> may be false, but both cannot be true. <i>Note 2:</i> If one is true the other one is a parent, not necessarily a direct parent.
<code>firstHalflife</code>	Half-life of the first nuclide in units <code>halflifeUnit</code> .
<code>uncFirstHalflife</code>	Absolute one sigma uncertainty of <code>firstHalflife</code> in units <code>halflifeUnit</code> .
<code>secondHalflife</code>	Half-life of the second nuclide in units <code>halflifeUnit</code> .
<code>uncSecondHalflife</code>	Absolute one sigma uncertainty of <code>secondHalflife</code> in units <code>halflifeUnit</code> .
<code>halflifeUnit</code>	Half-life unit. The supported units are year [y] or [a], day [d], hour [h], minute [m] or [min], and second [s].
<code>netBranching</code>	Net decay branching from parent to daughter through the decay chain, i.e., the product of branching fractions leading from parent to daughter.
<code>uncNetBranching</code>	One sigma absolute uncertainty of <code>netBranching</code> .
<code>refRatio</code>	Activity of the first nuclide divided by the activity of the second nuclide at <code>refTime</code> .
<code>uncRefRatio</code>	One sigma absolute uncertainty of <code>refRatio</code> .
<code>zeroRatio</code>	<i>A priori</i> assumed initial activity ratio between the first and the second nuclide at <code>zeroTime</code> . If not given, the nuclides must belong to the same decay chain, i.e., either <code>firstIsDaughter</code> or <code>secondIsDaughter</code> must be true. In that case it is assumed that the activity of the daughter is zero at <code>zeroTime</code> .
<code>uncZeroRatio</code>	One sigma absolute uncertainty of <code>zeroRatio</code> .
<code>refTime</code>	Reference date and time at which the <code>refRatio</code> is calculated. <i>Note:</i> If the half-lives are short when compared to the acquisition time, the decay corrections of activities to <code>refTime</code> should take the non-equilibrium of the decay chain into account. See Fig. 8.1.
<code>zeroTime</code>	Date and time when the initial activity was created.
<code>uncZeroTimeLow</code>	Uncertainty towards past of the <code>zeroTime</code> in hours [h].
<code>uncZeroTimeHigh</code>	Uncertainty towards future of the <code>zeroTime</code> in hours [h]. <i>Note:</i> Since the uncertainty of <code>zeroTime</code> is not necessarily symmetric, there are two uncertainties defined. The time together with the corresponding uncertainties is given as <code>zeroTime(-uncZeroTimeLow,+uncZeroTimeHigh)</code> , e.g., 2004-09-08 22:34:28 (-34.1h,+44.3h). The uncertainties correspond to one sigma absolute Gaussian uncertainties, i.e., 34% of the probability mass is between <code>zeroTime-uncZeroTimeLow</code> and <code>zeroTime</code> as well as between <code>zeroTime</code> and <code>zeroTime+uncZeroTimeHigh</code> .
<code>comments</code>	Comments in English [en].

8.7 Final Analysis Results

Table 8.8: Final analysis results

finalResults			
Field	Type	Length	Flags
idAnalysis	int	4	PFN
idMeas	int	4	IFN
idSample	int	4	IFN
completionTime	timestamp	4	
category	smallint	2	
categoryReason	text	0	
analyst	varchar	20	
analysisStatus	varchar	20	
purpose	text	0	
testType	text	0	
comparison	text	0	
conclusions	text	0	
projectFile	longblob	0	
lowQualitySpectrum	Boolean	1	
comments	text	0	

(idAnalysis) → analyses(idAnalysis)
(idMeas) → measurements(idMeas)
(idSample) → samples(idSample)

end of table.

idAnalysis	Identification number of the analysis. See table 8.1 analyses .
idMeas	Identification number of the measurement. See table 6.8 measurements .
idSample	Identification number of the sample measured. See table 6.1 samples .
completionTime	Date and time when these results were stored in the database.
category	Facility specific.
categoryReason	Motivations for <code>category</code> .
analyst	Name of the analyst who evaluated these results.
analysisStatus	Status of the analysis. Valid values are: <code>Preliminary</code> or <code>Final</code> .
purpose	Facility specific.
testType	Facility specific.
comparison	Facility specific.
conclusions	Facility specific.
projectFile	File containing the full project information. The contents are software specific. Typically this would be a script able to reconstruct all the final analysis results for the given sample and measurement. Alternatively this may be the name of the project file, or URI, where project information is to be found.
lowQualitySpectrum	Set TRUE if the analyst has deemed the spectrum as being of low quality.
comments	Comments in English [en].

Bibliography

- [1] UniSAMPO, Advanced Gamma Spectrum Analysis Software. User's Guide, Version 2.14, Doletum Oy, Ltd. Espoo, Finland. August 10, 2004,
- [2] SHAMAN – Expert System for Radionuclide Identification, version 1.13, User's Guide version 1.8, Baryon Oy, Ltd. Espoo, Finland. May 28, 2004.
- [3] <http://www.mysql.com>
- [4] *Linssi* – SQL Database for Gamma-Ray Spectrometry, Part II: SCRIPTS AND INTERFACES, Helsinki University of Technology, Espoo, Finland. (draft available upon request)
- [5] Le Système international d'unités (SI), 7e édition 1998, Organisation intergouvernementale de la Convention du Mètre, Stedi Paris, ISBN 92-822-2154-7.
- [6] Formats and Protocols for Messages, IDC-3.4.1 Revision 6, IDC Documentation.
- [7] User Manual of Radionuclide Analysis and Evaluation Software Aatami, version 3.04, Comprehensive Nuclear-Test-Ban Treaty Organization, Office of the Executive Secretary, Evaluation Section, Vienna, 2003.
- [8] Mathematical Markup Language (MathML) Version 2.0 (Second Edition), W3C Recommendation 21 October 2003, <http://www.w3.org/TR/2003/REC-MathML2-20031021/>
- [9] Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000, <http://www.w3.org/TR/2000/REC-xml-20001006>
- [10] XML Input for Expert System Shaman, User's Guide version 1.0, Baryon Oy, Ltd. Espoo, Finland. 2003.

Appendix A

Naming Conventions

The table and field (column) names are written in lower case, e.g. `stations`. If a name is composed of multiple words the words are separated by capitalizing their first letters, e.g. `sampleType`. To avoid excessive length the names may be abbreviated, e.g. `sampleCondFlagArrival`. There are special conventions applicable to keys only. Other fields are not allowed to use the syntax of keys. The rules are:

1. The keys of type integer have the prefix `id`, e.g. `idSample`.
2. The keys of non-integer type end with `Id`, e.g. `sampleId`.
3. The name of the foreign keys must be identical to the corresponding primary keys. There are some exceptions where this is not feasible:
 - Self-reference. A record may contain a key pointing to another record of the same table. In this case the key name is formed from the primary key by adding a prefix. E.g., the key `blankIdMeas` is this way formed from the primary key `idMeas`.
 - Multiple foreign keys. A record may have multiple foreign keys pointing to the same primary key. The prefixing is again used, e.g., two nuclides pointing to the primary key `nuclideId` are `firstNuclideId` and `secondNuclideId`.
4. The name of a unique non-integer field, with one-to-one correspondence with the primary integer key of the table, is formed by moving the prefix to the end of the name, e.g. from the primary `idSample` we get the name `sampleId` for the unique field. If the primary key and the unique field are both integers, or non-integers, the name of the unique field must be formed by prefixing. We see no reason for them being of the same type, however.

Note that in MySQL the names of the tables are case sensitive whereas the names of the fields are not. In *Linssi* the field names are unique regardless of case sensitivity. However, in order to be safe all the field and table names used must be typed following the conventions of this manual. In that way readability of the names is also enhanced.

