
DHIS2 implementation guide

DHIS2 Documentation Team



Copyright © 2008-2021 DHIS2 Team

Last update: 2021-12-01

Warranty: THIS DOCUMENT IS PROVIDED BY THE AUTHORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS MANUAL AND PRODUCTS MENTIONED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License: Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the source of this documentation, and is available here online: <http://www.gnu.org/licenses/fdl.html>

Table of contents

- A quick guide to DHIS2 implementation
 - Planning and organizing
 - Adapting DHIS2
 - Capacity building
- Conceptual Design Principles
 - All meta data can be added and modified through the user interface
 - A flexible data model supports different data sources to be integrated in one single data repository
 - Data input != Data output
 - Indicator-driven data analysis and reporting
 - Maintain disaggregated facility-data in the database
 - Support data analysis at any level in the health system
- Setting Up a New Database
 - Strategies for getting started
 - Controlled or open process?
 - Steps for developing a database
- Deployment Strategies
 - Offline Deployment
 - Online deployment
 - Hybrid deployment
 - Server hosting
- DHIS2 as Data Warehouse
 - Data warehouses and operational systems
 - Aggregation strategy in DHIS2
 - Data storage approach
- DHIS2 as a platform
 - Web portals
 - Apps
 - Information Systems
- Integration concepts
 - Integration and interoperability
 - Objectives of integration
 - Health information exchange
 - Aggregate and transactional data
 - Different DHIS2 integration scenarios
 - DHIS2 maturity model
 - Implementation steps for successful data and system integration
 - Specific integration and interoperability use cases
- Data Quality
 - Measuring data quality
 - Reasons for poor data quality
 - Improving data quality
 - Using DHIS2 to improve data quality
- Organisation Units
 - Organisation unit hierarchy design
 - Organisation unit groups and group sets
- Data Elements and Custom Dimensions
 - Data elements
 - Categories and custom dimensions
 - Data element groups
- Data Sets and Forms
 - What is a data set?

- [**What is a data entry form?**](#)
- [**From paper to electronic form - Lessons learned**](#)
- [**Indicators**](#)
 - [**What is an indicator?**](#)
 - [**Purpose of indicators**](#)
 - [**Indicator-driven data collection**](#)
 - [**Managing indicators**](#)
- [**Users and user roles**](#)
 - [**About user management**](#)
 - [**Workflow**](#)
 - [**Example: user management in a health system**](#)
- [**Guidelines for offline data entry using DHIS 2**](#)
 - [**Cases and corresponding solutions**](#)
 - [**Integrating tracker and aggregate data**](#)
 - [**Alternative approaches**](#)
 - [**Choosing an approach**](#)
 - [**How-to: saving aggregated tracker data as aggregate data values**](#)
- [**Data Analysis Tools Overview**](#)
 - [**Data analysis tools**](#)
- [**Localization of DHIS 2**](#)
 - [**DHIS 2 localization concepts**](#)
 - [**User Interface localization**](#)
 - [**Metadata/Database translations**](#)
- [**DHIS 2 Documentation Guide**](#)
 - [**DHIS 2 Documentation System Overview**](#)
 - [**Introduction**](#)
 - [**Getting started with GitHub**](#)
 - [**Getting the document source**](#)
 - [**Editing the documentation**](#)
 - [**DHIS 2 Bibliography**](#)
 - [**Handling multilingual documentation**](#)
 - [**Committing your changes back to GitHub**](#)
 - [**Markdown support and extensions**](#)
 - [**Submitting quick document fixes**](#)
 - [**Typo fix walk-through**](#)
- [**Using JIRA for DHIS2 issues**](#)
 - [**Sign up to JIRA - it's open to everyone!**](#)
 - [**Report an issue**](#)
 - [**Search for issues**](#)
 - [**About filters**](#)
 - [**Create a filter**](#)
 - [**Add a filter to your profile**](#)
 - [**Remove search filter terms from your search**](#)
 - [**Communicate with us**](#)
- [**Support**](#)
 - [**Home page: dhis2.org**](#)
 - [**Collaboration platform: community.dhis2.org**](#)
 - [**Reporting a problem**](#)
 - [**Development tracking: jira.dhis2.org**](#)
 - [**Source code: github.com/dhis2**](#)

A quick guide to DHIS2 implementation

Any implementation of District Health Information Software (DHIS2) should aim at establishing sustainable systems that are flexible to changing needs in the health sector. It is important to acknowledge that this will take many years, with continuous structures for capacity building, best practice sharing, and innovation. This quick guide will provide a very crude overview of the different facets of DHIS2 implementation.

Planning and organizing

Structures needed

- A DHIS core team (DCT) of 4-5 people will be needed to administer a national HMIS. Their responsibilities and required skills should be clearly defined. The DCT will participate in DHIS2 Academies, organize training and end-user support for various user groups in the country.
- A Technical Steering Committee, or equivalent, will be needed to steer the coordination between health programs, other information systems and development partners and Universities. They will lead integration efforts and make decisions regarding the overall architecture of information systems.

Integration efforts

- Throughout the implementation, simultaneous efforts of information system integration and data exchange need to be conducted. The leading principle for this work should be to create a decision-driven and indicator-focused system.

Equipment and internet

- An assessment needs to establish the needs for hardware. Desktops, laptops, tablets, mobile phones all have different qualities, and typically a mix of these different technologies will need to be supported.
- Server and hosting alternatives need to be critically examined with regards to capacity, infrastructural constraints, legal framework, security and confidentiality issues.
- Internet connection for all users will be needed. Mobile internet will be adequate for majority of users doing data collection and regular analysis.
- Options for mobile phone users, bulk sms deals etc, should be examined if appropriate.

Roll-out strategy

- The DCT will play a key role here and each member should have clear responsibilities for the roll-out covering: user support, user training, liaison with health programs, etc.
- Broader support structures need to be established to provide support, supervision, and communication with global/regional network of expert users and developers.
- Information use must be a focus area from the start and be a component both in the initial system design and the first round of user training. Data collection and data quality will only increase with real value of the information. District review meetings and equivalent should be supported with appropriate information products and training.
- Training will typically be the largest investment over time, and necessitates structures for continuous opportunities. Plan for a long term training approach catering for a continuous process of enabling new users and new system functionalities.
- Supervision and data quality assessment should be held frequently.

Adapting DHIS2

Scope of system

- Based on the decisions the system should support (system scope); customization and adaptation of the platform will be needed for DHIS2 aggregate, tracker, and/or events. Each action will need special competence, and should be led by the DCT.
- Assessment of the intended users and beneficiaries is needed, such as related to their information needs, and hardware and network needs.
- An understanding of the larger architecture of the HIS (the "HIS ecosystem") is important; what other systems are there, and how should they interact with DHIS2? Consider what needs there will be for interoperability between electronic systems.
- If there are needs that are not currently supported by DHIS2, an assessment of additional software development is necessary. These can be addressed locally by developing a custom web app or feed into the overall core platform development roadmap process organized by UiO.

Setting up DHIS2

- Reporting units: implementing the different reporting units (service outlets) and hierarchies including grouping.
- Data collection needs: Which indicators are needed, what data variables will go into their calculation, and how should this data be collected? Design data elements, disaggregation categories, data sets, and collection forms.
- Information for action (indicators, dashboards, other outputs): what are the information products the various users will need? Tables, charts, maps, dashboards. Routines for dissemination and sharing.
- User management: Create user roles and groups, routines for managing users, define access to features, and appropriate sharing of content.
- DHIS2 governance document (roles by profile, how to change metadata and under what conditions).

Hosting

- There are many different options for hosting an online system, both in terms of where to put the server (e.g. in-house vs. cloud) and who to manage the server (e.g. in-house vs. outsourced). Server and hosting alternatives needs to be critically examined with regards to capacity, infrastructural constraints, legal framework, security and confidentiality issues. These decisions may need to be revisited at least annually as server complexity, data types (e.g. aggregate vs. patient) and local capacity may change over time.

Capacity building

DHIS core team (DCT)

- DCT will need all skills necessary for a sustainable, evolving system. This includes technical skills (DHIS2 adaptation, server maintenance), system knowledge (architectures and design principles), organizational (integration strategies), and project management (organizing structured support and training).
- DCT members should attend the regional/global DHIS2 Academy frequently (e.g. twice a year) to ensure high quality training, continuous communication with the broader expert community, and to make sure the local team is up to date with new functionalities and enhancements in

recent releases of the DHIS 2 platform. DCT will be responsible for adapting and cascading this regional training curriculum to a broader group of users within country.

Country training strategies

- DCT should offer training in relation to the implementation, and continuously thereafter to meet growing demands, system updates and staff turnover.
- Adapting and developing training material and reference guides to reflect local information needs and local system content is important.

Continuous training opportunities

- As user experience is growing, more advanced training should be offered. Information use training for district medical officers and health program managers is crucial early on to enroll stakeholders to use the information in decision making.

Conceptual Design Principles

This chapter provides a introduction to some of the key conceptual design principles behind the DHIS2 software. Understanding and being aware of these principles will help the implementer to make better use of the software when customising a local database. While this chapter introduces the principles, the following chapters will detail out how these are reflected in the database design process.

The following conceptual design principles will be presented in this chapter:

- All meta data can be added and modified through the user interface
- A flexible data model supports different data sources to be integrated in one single data repository
- Data Input != Data Output
- Indicator-driven data analysis and reporting
- Maintain disaggregated facility-data in the database
- Support data analysis at any level in the health system

In the following section each principle is described in more detail.

All meta data can be added and modified through the user interface

The DHIS2 application comes with a set of generic tools for data collection, validation, reporting and analysis, but the contents of the database, e.g. what data to collect, where the data comes from, and on what format, will depend on the context of use. These meta data need to be populated into the application before it can be used, and this can be done through the user interface and requires no programming. This allows for more direct involvement of the domain experts that understand the details of the HIS that the software will support.

The software separates the key meta data that describes the raw data being stored in the database, which is the critical meta data that should not change much over time (to avoid corrupting the data), and the higher level meta like indicator formulas, validation rules, and groups for aggregation as well as the various layouts for collection forms and reports, which are not that critical and can be changed over time without interfering with the raw data. As this higher level meta data can be added and modified over time without interfering with the raw data, a continuous customisation process is supported. Typically new features are added over time as the local implementation team learn to master more functionality, and the users are gradually pushing for more advanced data analysis and reporting outputs.

A flexible data model supports different data sources to be integrated in one single data repository

The DHIS2 design follows an integrated approach to HIS, and supports integration of many different data sources into one single database, sometime referred to as an integrated data repository or a data warehouse.

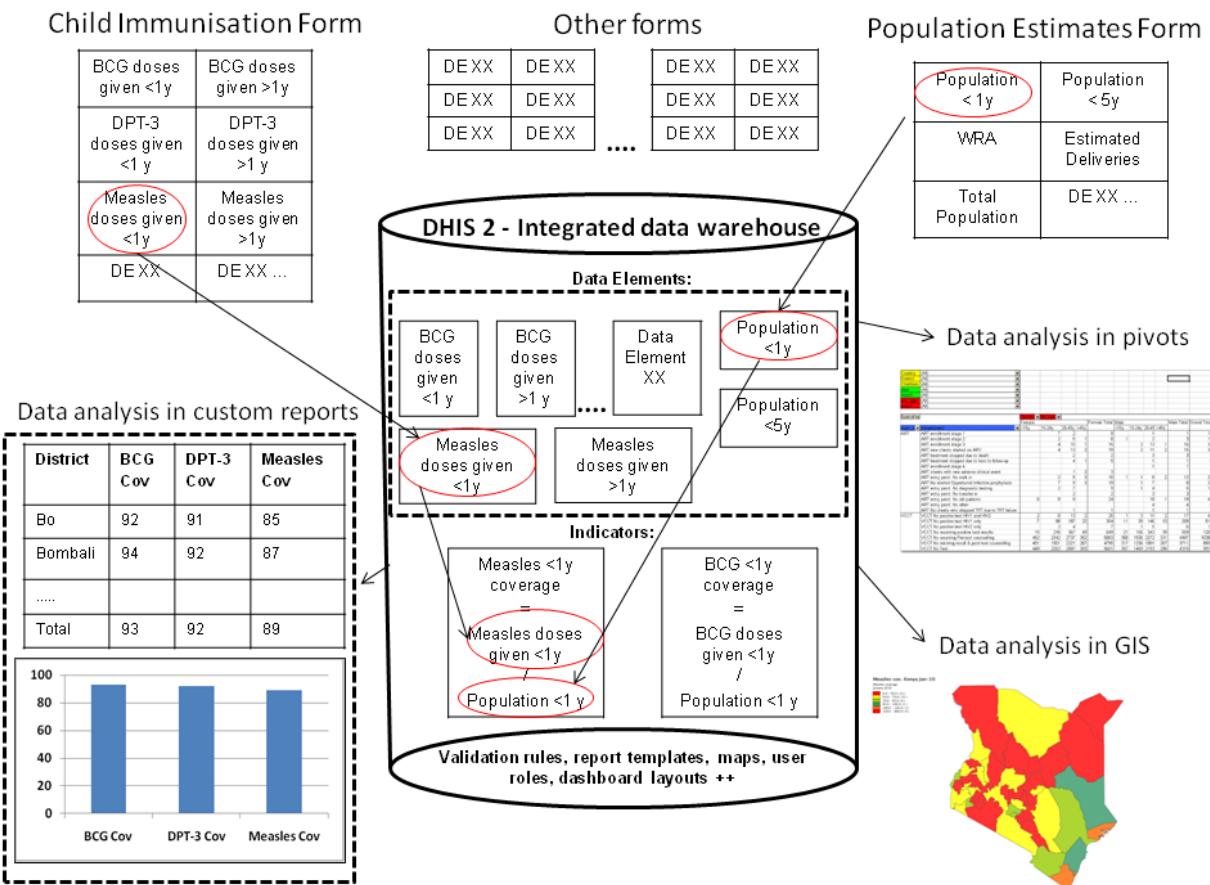
The fact that DHIS2 is a skeleton like tool without predefined forms or reports means that it can support a lot of different aggregate data sources. There is nothing really that limits the use to the health domain either, although use in other sectors are still very limited. As long as the data is collected by an orgunit (organisational unit), described as a data element (possibly with some disaggregation categories), and can be represented by a predefined period frequency, it can be collected and processed in DHIS2. This flexibility makes DHIS2 a powerful tool to set up integrated systems that bring together collection tools, indicators, and reports from multiple health programs, departments or initiatives. Once the data is defined and then collected or imported into a DHIS2

database, it can be analysed in correlation to any other data in the same database, no matter how and by whom it was collected. In addition to supporting integrated data analysis and reporting, this integrated approach also helps to rationalise data collection and reduce duplication.

Data input != Data output

In DHIS2 there are three dimensions that describe the aggregated data being collected and stored in the database; the where - organisation unit, the what - data element, and the when - period. The organisation unit, data element and period make up the three core dimensions that are needed to describe any data value in the DHIS2, whether it is in a data collection form, a chart, on a map, or in an aggregated summary report. When data is collected in an electronic data entry form, sometimes through a mirror image of the paper forms used at facility level, each entry field in the form can be described using these three dimensions. The form itself is just a tool to organise the data collection and is not describing the individual data values being collected and stored in the database. Being able to describe each data value independently through a Data Element definition (e.g. 'Measles doses given <1 year') provides important flexibility when processing, validating, and analysing the data, and allows for comparison of data across collection forms and health programs.

This design or data model approach separates DHIS2 from many of the traditional HIS software applications which treat the data collection forms as the key unit of analysis. This is typical for systems tailored to vertical programs' needs and the traditional conceptualisation of the collection form as also being the report or the analysis output. The figure below illustrates how the more fine-grained DHIS2 design built around the concept of Data Elements is different and how the input (data collection) is separated from the output (data analysis), supporting more flexible and varied data analysis and dissemination. The data element 'Measles doses given <1 y' is collected as part of a Child Immunisation collection form, but can be used individually to build up an Indicator (a formula) called 'Measles coverage <1y' where it is combined with the data element called 'Population <1y', being collected through another collection form. This calculated Indicator value can then be used in data analysis in various reporting tools in DHIS2, e.g. custom designed reports with charts, pivot tables, or on a map in the GIS module.



Indicator-driven data analysis and reporting

What is referred to as a Data Element above, the key dimension that describes what is being collected, is sometimes referred to as an indicator in other settings. In DHIS2 we distinguish between Data Elements which describe the raw data, e.g. the counts being collected, and Indicators, which are formula-based and describe calculated values, e.g. coverage or incidence rates that are used for data analysis. Indicator values are not collected like the data (element) values, but instead calculated by the application based on formulas defined by the users. These formulas are made up of a factor (e.g. 1, 100, 100, 100 000), a numerator and a denominator, the two latter are both expressions based on one or more data elements. E.g. the indicator "Measles coverage <1 year" is defined a formula with a factor 100, a numerator ("Measles doses given to children under 1 year") and a denominator ("Target population under 1 year"). The indicator "DPT1 to DPT3 drop out rate" is a formula of 100 % x ("DPT1 doses given"- "DPT3doses given") / ("DPT1 doses given"). These formulas can be added and edited through the user interface by a user with limited training, as they are quite easy to set up and do not interfere with the data values stored in the database (so adding or modifying an indicator is not a critical operation).

Indicators represent perhaps the most powerful data analysis feature of the DHIS2, and all reporting tools support the use of indicators, e.g. as displayed in the custom report in the figure above. Being able to use population data in the denominator enables comparisons of health performance across geographical areas with different target populations, which is more useful than only looking at the raw numbers. The table below uses both the raw data values (Doses) and indicator values (Cov) for the different vaccines. Comparing e.g. the two first orgunits in the list, Taita Taveta County and Kilifi County, on DPT-1 immunisation, we can see that while the raw numbers (659 vs 2088) indicate many more doses are given in Kilifi, the coverage rates (92.2 % vs 47.5 %) show that Taita Taveta are doing a better job immunising their target population under 1 year. Looking at the final column (Immuniz. Compl. %) which indicates the completeness of reporting of the immunisation form for the same period, we can see that the numbers are more or less the same in the two counties we compared, which tells us that the coverage rates can be reasonably compared across the two counties.

Organisation	DPT 1		DPT 2		DPT 3		Measles		Fully Imm		Immuniz Compl %
	Doses	Cov	Doses	Cov	Doses	Cov	Doses	Cov	Doses	Cov	
Taita Taveta County	659	92.2	630	89.1	580	81.0	574	80.2	551	77.4	81.4
Kilifi County	2088	47.5	1767	39.6	1954	43.0	3315	73.4	2540	55.5	82.1
Lamu County	238	82.6	200	70.0	268	91.4	283	97.7	195	60.6	81.0
Kwale County	1142	51.1	1077	48.3	1149	52.4	1909	86.3	1385	62.2	84.1
Mombasa County	2015	73.2	1711	62.4	1948	70.7	2737	98.0	2278	82.2	88.4
Tana River County	678	80.1	633	77.4	721	76.2	656	79.6	404	50.0	78.3
Coast	6820	60.6	6018	53.5	6620	57.7	9474	83.5	7353	64.6	83.2

Maintain disaggregated facility-data in the database

When data is collected and stored in DHIS2 it will remain disaggregated in the database with the same level of detail as it was collected. This is a major advantage of having a database system for HIS as supposed to a paper-based or even spreadsheet based system. The system is designed to store large amounts of data and always allow drill-downs to the finest level of detail possible, which is only limited by how the data was collected or imported into the DHIS2 database. In a perspective of a national HIS it is desired to keep the data disaggregated by health facility level, which is often the lowest level in the orgunit hierarchy. This can be done even without computerising this level, through a hybrid system of paper and computer. The data can be submitted from health facilities to e.g. district offices by paper (e.g. on monthly summary forms for one specific facility), and then at the district office they enter all the facility data into the DHIS2 through the electronic data collection forms, one facility at a time. This will enable the districts health management teams to perform facility-wise data analysis and to e.g. provide print-outs of feedback reports generated by the DHIS2, incl. facility comparisons, to the facility in-charges in their district.

Support data analysis at any level in the health system

While the name DHIS2 indicates a focus on the District, the application provides the same tools and functionality to all levels in the health system. In all the reporting tools the users can select which orgunit or orgunit level to analyse and the data displayed will be automatically aggregated up to the selected level. The DHIS2 uses the orgunit hierarchy in aggregating data upwards and provides data by any orgunit in this hierarchy. Most of the reports are run in such a way that the users will be prompted to select an orgunit and thereby enable reuse of the same report layouts for all levels. Or if desired, the report layouts can be tailored to any specific level in the health system if the needs differ between the levels.

In the GIS module the users can analyse data on e.g. the sub-national level and then by clicking on the map (on e.g. a region or province) drill down to the next level, and continue like this all the way

down to the source of the data at facility level. Similar drill-down functionality is provided in the Excel Pivot Tables that are linked to the DHIS2 database.

To speed up performance and reduce the response-time when providing aggregated data outputs, which may include many calculations (e.g. adding together 8000 facilities), DHIS2 pre-calculates all the possible aggregate values and stores these in what is called a data mart. This data mart can be scheduled to run (re-built) at a given time interval, e.g. every night.

Setting Up a New Database

The DHIS2 application comes with a set of tools for data collection, validation, reporting and analysis, but the contents of the database, e.g. what data to collect, where the data comes from, and on what format will depend on the context of use. These meta data need to be populated into the application before it can be used, and this can be done through the user interface and requires no programming. What is required is in-depth knowledge about the local HIS context as well as an understanding of the DHIS2 design principles (see the chapter “Key conceptual design principles in DHIS2”). We call this initial process database design or customisation. This chapter provides an overview of the customisation process and briefly explains the steps involved, in order to give the implementer a feeling of what this process requires. Other chapters in this manual provide a lot more detail into some of the specific steps.

Strategies for getting started

The following section describes a list of tips for getting off to a good start when developing a new database.

1. Quickly populate a demo database, incl. examples of reports, charts, dashboard, GIS, data entry forms. Use real data, ideally nation-wide, but not necessarily facility-level data.
2. Put the demo database online. Server hosting with an external provider server can be a solution to speed up the process, even if temporary. This makes a great collaborative platform and dissemination tool to get buy-in from stakeholders.
3. The next phase is a more elaborate database design process. Parts of the demo can be reused if viable.
4. Make sure to have a local team with different skills and background: public health, data administrator, IT and project management.
5. Use the customisation and database design phase as a learning and training process to build local capacity through learning-by-doing.
6. The country national team should drive the database design process but be supported and guided by experienced implementers.

Controlled or open process?

As the DHIS2 customisation process often is and should be a collaborative process, it is also important to have in mind which parts of the database are more critical than others, e.g. to avoid an untrained user to corrupt the data. Typically it is a lot more critical to customise a database which already has data values, than working with meta data on an “empty” database. Although it might seem strange, much customisation takes place after the first data collection or import has started, e.g. when adding new validation rules, indicators or report layouts. The most critical mistake that can be made is to modify the meta data that directly describes the data values, and these as we have seen above, are the *data elements* and the *organisation units*. When modifying these definitions it is important to think about how the change will affect the meaning of the data values already in the system (collected using the old definitions). It is recommended to limit who can edit these core meta data through the user role management, to restrict the access to a core customisation team.

Other parts of the system that are not directly coupled to the data values are a lot less critical to play around with, and here, at least in the early phases, one should encourage the users to try out new things in order to create learning. This goes for groups, validation rules, indicator formulas, charts, and reports. All these can easily be deleted or modified later without affecting the underlying data values, and therefore are not critical elements in the customisation process.

Of course, later in the customisation process when going into a production phase, one should be even more careful in allowing access to edit the various meta data, as any change, also to the less critical meta data, might affect how data is aggregated together or presented in a report (although the underlying raw data is still safe and correct).

Steps for developing a database

The following section describes concrete steps for developing a database from scratch.

The organisational hierarchy

The organisational hierarchy defines the organisation using the DHIS2, the health facilities, administrative areas and other geographical areas used in data collection and data analysis. This dimension to the data is defined as a hierarchy with one root unit (e.g. Ministry of Health) and any number of levels and nodes below. Each node in this hierarchy is called an organisational unit in DHIS2. The design of this hierarchy will determine the geographical units of analysis available to the users as data is collected and aggregated in this structure. There can only be one organisational hierarchy at the same time so its structure needs careful consideration.

Additional hierarchies (e.g. parallel administrative boundaries to the health care sector) can be modelled using organisational groups and group sets, but the organisational hierarchy is the main vehicle for data aggregation on the geographical dimension. Typically national organisational hierarchies in public health have 4-6 levels, but any number of levels is supported. The hierarchy is built up of parent-child relations, e.g. a Country or MoH unit (the root) might have e.g. 8 child units (provinces), and each province again (at level 2) might have 10-15 districts as their children. Normally the health facilities will be located at the lowest level, but they can also be located at higher levels, e.g. national or provincial hospitals, so skewed organisational trees are supported (e.g. a leaf node can be positioned at level 2 while most other leaf nodes are at level 5).

Data Elements

The Data Element is perhaps the most important building block of a DHIS2 database. It represents the *what* dimension, it explains what is being collected or analysed. In some contexts, this is referred to an indicator, but in DHIS2 we call this unit of collection and analysis a data element. The data element often represents a count of something, and its name describes what is being counted, e.g. "BCG doses given" or "Malaria cases". When data is collected, validated, analysed, reported or presented it is the data elements or expressions built upon data elements that describe the *WHAT* of the data. As such the data elements become important for all aspects of the system and they decide not only how data is collected, but more importantly how the data values are represented in the database, which again decides how data can be analysed and presented.

A best practice when designing data elements is to think of data elements as a unit of data analysis and not just as a field in the data collection form. Each data element lives on its own in the database, completely detached from the collection form, and reports and other outputs are based on data elements and expressions/formulas composed of data elements and not the data collection forms. So the data analysis needs should drive the process, and not the look and feel of the data collection forms.

Data sets and data entry forms

All data entry in DHIS2 is organised through the use of data sets. A data set is a collection of data elements grouped together for data collection, and in the case of distributed installs they also define chunks of data for export and import between instances of DHIS2 (e.g. from a district office local installation to a national server). Data sets are not linked directly to the data values, only through their data elements and frequencies, and as such a data set can be modified, deleted or added at any point in time without affecting the raw data already captured in the system, but such changes will of course affect how new data will be collected.

Once you have assigned a data set to an organisation unit that data set will be made available in Data Entry (under Services) for the organisation units you have assigned it to and for the valid periods according to the data set's period type. A default data entry form will then be shown, which is simply a list of the data elements belonging to the data set together with a column for inputting the values. If your data set contains data elements with categories such as age groups or gender, then additional columns will be automatically generated in the default form based on the categories. In addition to the default list-based data entry form, there are two more alternatives, the section-based form and the custom form. Section forms allow for a bit more flexibility when it comes to using tabular forms and are quick and simple to design. Often your data entry form will need multiple tables with subheadings, and sometimes you need to disable (grey out) a few fields in the table (e.g. some categories do not apply to all data elements), both of these functions are supported in section forms. When the form you want to design is too complicated for the default or section forms then your last option is to use a custom form. This takes more time but gives you full flexibility in term of the design. In DHIS2 there is a built in HTML editor (FcK Editor) for the form designer and you can either design the form in the UI or paste in your html directly (using the Source window in the editor).

Validation rules

Once you have set up the data entry part of the system and started to collect data then there is time to define data quality checks that help to improve the quality of the data being collected. You can add as many validation rules as you like and these are composed of left and right side expressions that again are composed of data elements, with an operator between the two sides. Typical rules are comparing subtotals to totals of something. E.g. if you have two data elements "HIV tests taken" and "HIV test result positive" then you know that in the same form (for the same period and organisational unit) the total number of tests must always be equal or higher than the number of positive tests. These rules should be absolute rules meaning that they are mathematically correct and not just assumptions or "most of the time correct". The rules can be run in data entry, after filling each form, or as a more batch like process on multiple forms at the same time, e.g. for all facilities for the previous reporting month. The results of the tests will list all violations and the detailed values for each side of the expression where the violation occurred to make it easy to go back to data entry and correct the values.

Indicators

Indicators represent perhaps the most powerful data analysis feature of the DHIS2. While data elements represent the raw data (counts) being collected the indicators represent formulas providing coverage rates, incidence rates, ratios and other formula-based units of analysis. An indicator is made up of a factor (e.g. 1, 100, 100, 100 000), a numerator and a denominator, the two latter are both expressions based on one or more data elements. E.g. the indicator "BCG coverage <1 year" is defined a formula with a factor 100, a numerator ("BCG doses given to children under 1 year") and a denominator ("Target population under 1 year"). The indicator "DPT1 to DPT3 drop out rate" is a formula of 100 % x ("DPT1 doses given"- "DPT3 doses given") / ("DPT1 doses given").

Most report modules in DHIS2 support both data elements and indicators and you can also combine these in custom reports, but the important difference and strength of indicators versus raw data (data element's data values) is the ability to compare data across different geographical areas (e.g. highly populated vs rural areas) as the target population can be used in the denominator.

Indicators can be added, modified and deleted at any point in time without interfering with the data values in the database.

Report tables and reports

Standard reports in DHIS2 is a very flexible way of presenting the data that has been collected. Data can be aggregated by any organisational unit or orgunit level, by data element, by indicators, as well as over time (e.g. monthly, quarterly, yearly). The report tables are custom data sources for the standard reports and can be flexibly defined in the user interface and later accessed in external report

designers such as iReport or BIRT. These report designs can then be set up as easily accessible one-click reports with parameters so that the users can run the same reports e.g. every month when new data is entered, and also be relevant to users at all levels as the organisational unit can be selected at the time of running the report.

GIS (Maps)

In the integrated GIS module you can easily display your data on maps, both on polygons (areas) and as points (health facilities), and either as data elements or indicators. By providing the coordinates of your organisational units to the system you can quickly get up to speed with this module. See the GIS section for details on how to get started.

Charts and dashboard

One of the easiest way to display your indicator data is through charts. An easy to use chart dialogue will guide you through the creation of various types of charts with data on indicators, organisational units and periods of your choice. These charts can easily be added to one of the four chart sections on your dashboard and there be made easily available right after log in. Make sure to set the dashboard module as the start module in user settings.

Deployment Strategies

DHIS2 is a network enabled application and can be accessed over the Internet, a local intranet and as a locally installed system. The deployment alternatives for DHIS2 are in this chapter defined as i) offline deployment ii) online deployment and iii) hybrid deployment. The meaning and differences will be discussed in the following sections.

Offline Deployment

An offline deployment implies that multiple standalone offline instances are installed for end users, typically at the district level. The system is maintained primarily by the end users/district health officers who enters data and generate reports from the system running on their local server. The system will also typically be maintained by a national super-user team who pay regular visits to the district deployments. Data is moved upwards in the hierarchy by the end users producing data exchange files which are sent electronically by email or physically by mail or personal travel. (Note that the brief Internet connectivity required for sending emails does not qualify for being defined as online). This style of deployment has the obvious benefit that it works when appropriate Internet connectivity is not available. On the other side there are significant challenges with this style which are described in the following section.

- **Hardware:** Running stand-alone systems requires advanced hardware in terms of servers and reliable power supply to be installed, usually at district level, all over the country. This requires appropriate funding for procurement and plan for long-term maintenance.
- **Software platform:** Local installs implies a significant need for maintenance. From experience, the biggest challenge is viruses and other malware which tend to infect local installations in the long-run. The main reason is that end users utilize memory sticks for transporting data exchange files and documents between private computers, other workstations and the system running the application. Keeping anti-virus software and operating system patches up to date in an offline environment are challenging and bad practices in terms of security are often adopted by end users. The preferred way to overcome this issue is to run a dedicated server for the application where no memory sticks are allowed and use a Linux based operating system which is not as prone to virus infections as MS Windows.
- **Software application:** Being able to distribute new functionality and bug-fixes to the health information software to users are essential for maintenance and improvement of the system. Relying on the end users to perform software upgrades requires extensive training and a high level of competence on their side as upgrading software applications might be a technically challenging task. Relying on a national super-user team to maintain the software implies a lot of travelling.
- **Database maintenance:** A prerequisite for an efficient system is that all users enter data with a standardized meta-data set (data elements, forms etc). As with the previous point about software upgrades, distribution of changes to the meta-data set to numerous offline installations requires end user competence if the updates are sent electronically or a well-organized super-user team. Failure to keep the meta-data set synchronized will lead to loss of ability to move data from the districts and/or an inconsistent national database since the data entered for instance at the district level will not be compatible with the data at the national level.

Online deployment

An online deployment implies that a single instance of the application is set up on a server connected to the Internet. All users (clients) connect to the online central server over the Internet using a web browser. This style of deployment currently benefits from the huge investments in and expansions of mobile networks in developing countries. This makes it possible to access online servers in even the most rural areas using mobile Internet modems (also referred to as *dongles*).

This online deployment style has huge positive implications for the implementation process and application maintenance compared to the traditional offline standalone style:

- Hardware: Hardware requirements on the end-user side are limited to a reasonably modern computer/laptop and Internet connectivity through a fixed line or a mobile modem. There is no need for a specialized server, any Internet enabled computer will be sufficient.
- Software platform: The end users only need a web browser to connect to the online server. All popular operating systems today are shipped with a web browser and there is no special requirement on what type or version. This means that if severe problems such as virus infections or software corruption occur one can always resort to re-formatting and installing the computer operating system or obtain a new computer/laptop. The user can continue with data entry where it was left and no data will be lost.
- Software application: The central server deployment style means that the application can be upgraded and maintained in a centralized fashion. When new versions of the applications are released with new features and bug-fixes it can be deployed to the single online server. All changes will then be reflected on the client side the next time end users connect over the Internet. This obviously has a huge positive impact for the process of improving the system as new features can be distributed to users immediately, all users will be accessing the same application version, and bugs and issues can be sorted out and deployed on-the-fly.
- Database maintenance: Similar to the previous point, changes to the meta-data can be done on the online server in a centralized fashion and will automatically propagate to all clients next time they connect to the server. This effectively removes the vast issues related to maintaining an upgraded and standardized meta-data set related to the traditional offline deployment style. It is extremely convenient for instance during the initial database development phase and during the annual database revision processes as end users will be accessing a consistent and standardized database even when changes occur frequently.

This approach might be problematic in cases where Internet connectivity is volatile or missing in long periods of time. DHIS2 however has certain features which require Internet connectivity to be available only part of the time for the system to work properly, such as the MyDatamart tool presented in a separate chapter in this guide.

Hybrid deployment

From the discussion so far one realizes that the online deployment style is favourable over the offline style but requires decent Internet connectivity where it will be used. It is important to notice that the mentioned styles can co-exist in a common deployment. It is perfectly feasible to have online as well as offline deployments within a single country. The general rule would be that districts and facilities should access the system online over the Internet where sufficient Internet connectivity exist, and offline systems should be deployed to districts where this is not the case.

Defining decent Internet connectivity precisely is hard but as a rule of thumb the download speed should be minimum 10 Kbyte/second and accessibility should be minimum 70% of the time.

In this regard, mobile Internet modems which can be connected to a computer or laptop and access the mobile network are an extremely capable and feasible solution. Mobile Internet coverage is increasing rapidly all over the world, often provides excellent connectivity at low prices and is a great alternative to local networks and poorly maintained fixed Internet lines. Getting in contact with national mobile network companies regarding post-paid subscriptions and potential large-order benefits can be a worthwhile effort. The network coverage for each network operator in the relevant country should be investigated when deciding which deployment approach to opt for as it might differ and cover different parts of the country.

Server hosting

The online deployment approach raises the question of where and how to host the server which will run the DHIS2 application. Typically there are several options:

1. Internal hosting within the Ministry of Health
2. Hosting within a government data centre
3. Hosting through an external hosting company

The main reason for choosing the first option is often political motivation for having “physical ownership” of the database. This is perceived as important by many in order to “own” and control the data. There is also a wish to build local capacity for server administration related to sustainability of the project. This is often a donor-driven initiative as it is perceived as a concrete and helpful mission.

Regarding the second option, some places a government data centre is constructed with a view to promoting and improving the use and accessibility of public data. Another reason is that a proliferation of internal server environments is very resource demanding and it is more effective to establish centralized infrastructure and capacity.

Regarding external hosting, there is lately a move towards outsourcing the operation and administration of computer resources to an external provider, where those resources are accessed over the network, popularly referred to as “cloud computing” or “software as a service”. Those resources are typically accessed over the Internet using a web browser.

The primary goal for an online server deployment is to provide long-term stable and high-performance accessibility to the intended services. When deciding which option to choose for server environment there are many aspects to consider:

1. Human capacity for server administration and operation. There must be human resources with general skills in server administration and in the specific technologies used for the application providing the services. Examples of such technologies are web servers and database management platforms.
2. Reliable solutions for automated backups, including local off-server and remote backup.
3. Stable connectivity and high network bandwidth for traffic to and from the server.
4. Stable power supply including a backup solution.
5. Secure environment for the physical server regarding issues such as access, theft and fire.
6. Presence of a disaster recovery plan. This plan must contain a realistic strategy for making sure that the service will be only suffering short down-times in the events of hardware failures, network downtime and more.
7. Feasible, powerful and robust hardware.

All of these aspects must be covered in order to create an appropriate hosting environment. The hardware requirement is deliberately put last since there is a clear tendency to give it too much attention.

Looking back at the three main hosting options, experience from implementation missions in developing countries suggests that all of the hosting aspects are rarely present in option one and two at a feasible level. Reaching an acceptable level in all these aspects is challenging in terms of both human resources and money, especially when compared to the cost of option three. It has the benefit that it accommodates the mentioned political aspects and building local capacity for server administration, on the other hand, can this be provided for in alternative ways.

Option three - external hosting - has the benefit that it supports all of the mentioned hosting aspects at a very affordable price. Several hosting providers - of virtual servers or software as a service - offer reliable services for running most kinds of applications. Examples of such providers are Linode and Amazon Web Services. Administration of such servers happens over a network connection, which most often anyway is the case with local server administration. The physical location of the server in this case becomes irrelevant in that such providers offer services in most parts of the world. This solution is increasingly becoming the standard solution for hosting of application services. The aspect of building local capacity for server administration is compatible with this option since a local ICT team can be tasked with maintaining the externally hosted server.

An approach for combining the benefits of external hosting with the need for local hosting and physical ownership is to use an external hosting provider for the primary transactional system while mirroring this server to a locally hosted non-critical server which is used for read-only purposes such as data analysis and accessed over the intranet.

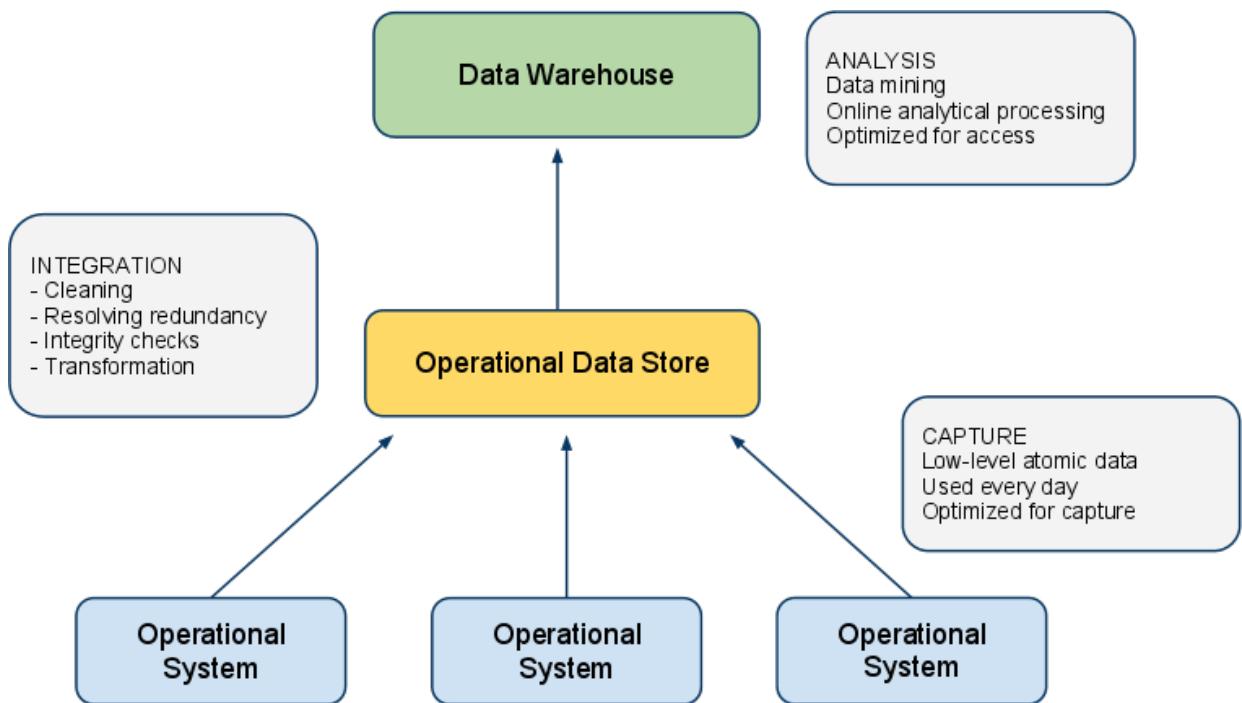
DHIS2 as Data Warehouse

This chapter will discuss the role and place of the DHIS2 application in a system architecture context. It will show that DHIS2 can serve the purpose of both a data warehouse and an operational system.

Data warehouses and operational systems

A *data warehouse* is commonly understood as a database used for analysis. Typically data is uploaded from various operational / transactional systems. Before data is loaded into the data warehouse it usually goes through various stages where it is cleaned for anomalies and redundancy and transformed to conform with the overall structure of the integrated database. Data is then made available for use by analysis, also known under terms such as **data mining** and *online analytical processing*. The data warehouse design is optimized for speed of data retrieval and analysis. To improve performance the data storage is often redundant in the sense that the data is stored both in its most granular form and in an aggregated (summarized) form.

A *transactional system* (or *operational system* from a data warehouse perspective) is a system that collects, stores and modifies low level data. This system is typically used on a day-to-day basis for data entry and validation. The design is optimized for fast insert and update performance.



There are several benefits of maintaining a data warehouse, some of them being:

- *Consistency*: It provides a common data model for all relevant data and acts as an abstraction over a potentially high number of data sources and feeding systems which makes it a lot easier to perform analysis.
- *Reliability*: It is detached from the sources where the data originated from and is hence not affected if data in the operational systems are purged or lost.
- *Analysis performance*: It is designed for maximum performance for data retrieval and analysis in contrast to operational systems which are often optimized for data capture.

There are however also significant challenges with a data warehouse approach:

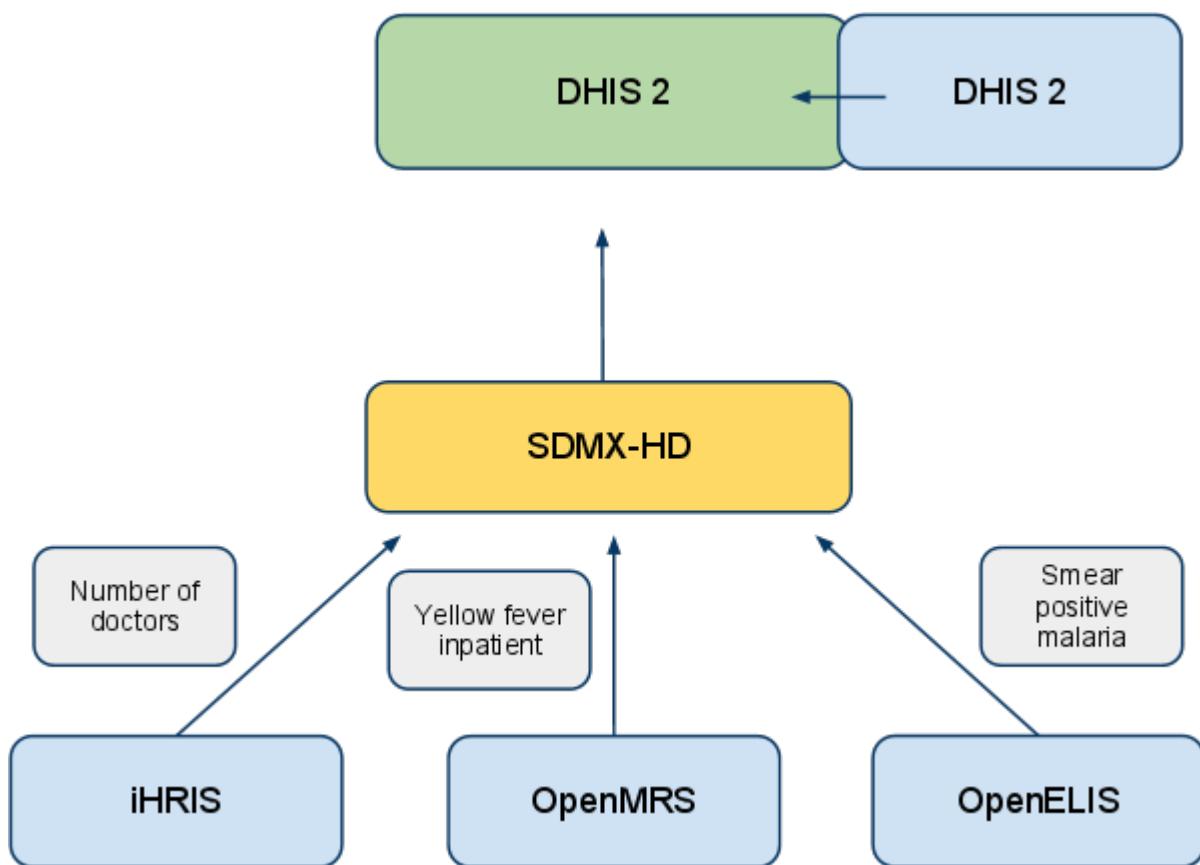
- *High cost*: There is a high cost associated with moving data from various sources into a common data warehouse, especially when the operational systems are not similar in nature. Often long-term existing systems (referred to as legacy systems) put heavy constraints on the data transformation process.
- *Data validity*: The process of moving data into the data warehouse is often complex and hence often not performed at regular and timely intervals. This will then leave the data users with outdated and irrelevant data not suitable for planning and informed decision making.

Due to the mentioned challenges, it has lately become increasingly popular to merge the functions of the data warehouse and operational system, either into a single system which performs both tasks or with tightly integrated systems hosted together. With this approach, the system provides functionality for data capture and validation as well as data analysis and manages the process of converting low-level atomic data into aggregate data suitable for analysis. This sets high standards for the system and its design as it must provide appropriate performance for both of those functions; however, advances in hardware and parallel processing is increasingly making such an approach feasible.

In this regard, the DHIS2 application is designed to serve as a tool for both data capture, validation, analysis and presentation of data. It provides modules for all of the mentioned aspects, including data entry functionality and a wide array of analysis tools such as reports, charts, maps, pivot tables and dashboard.

In addition, DHIS2 is a part of a suite of interoperable health information systems which covers a wide range of needs and are all open-source software. DHIS2 implements the standard for data and meta-data exchange in the health domain called SDMX-HD. There are many examples of operational systems which also implements this standard and potentially can feed data into DHIS2:

- iHRIS: System for management of human resource data. Examples of data which is relevant for a national data warehouse captured by this system is "number of doctors", "number of nurses" and "total number of staff". This data is interesting to compare for instance to district performance.
- OpenMRS: Medical record system being used at hospital. This system can potentially aggregate and export data on inpatient diseases to a national data warehouse.
- OpenELIS: Laboratory enterprise information system. This system can generate and export data on number and outcome of laboratory tests.



Aggregation strategy in DHIS2

The analysis tools in DHIS2 read aggregated data from *data mart* tables. A data mart is a data store optimized for meeting the most common user requests for data analysis. The DHIS2 data mart contains data aggregated in the *space dimension* (the organisation unit hierarchy), *time dimension* (over multiple periods) and for *indicator formulas* (mathematical expressions including data elements). Retrieving data directly from data marts provides good performance even in high-concurrency environments since most requests for analysis can be served with a single, simple database query against the data mart. The aggregation engine in DHIS2 is capable of processing low-level data in the multi-millions and managing most national-level databases, and it can be said to provide *near real-time* access to aggregate data.

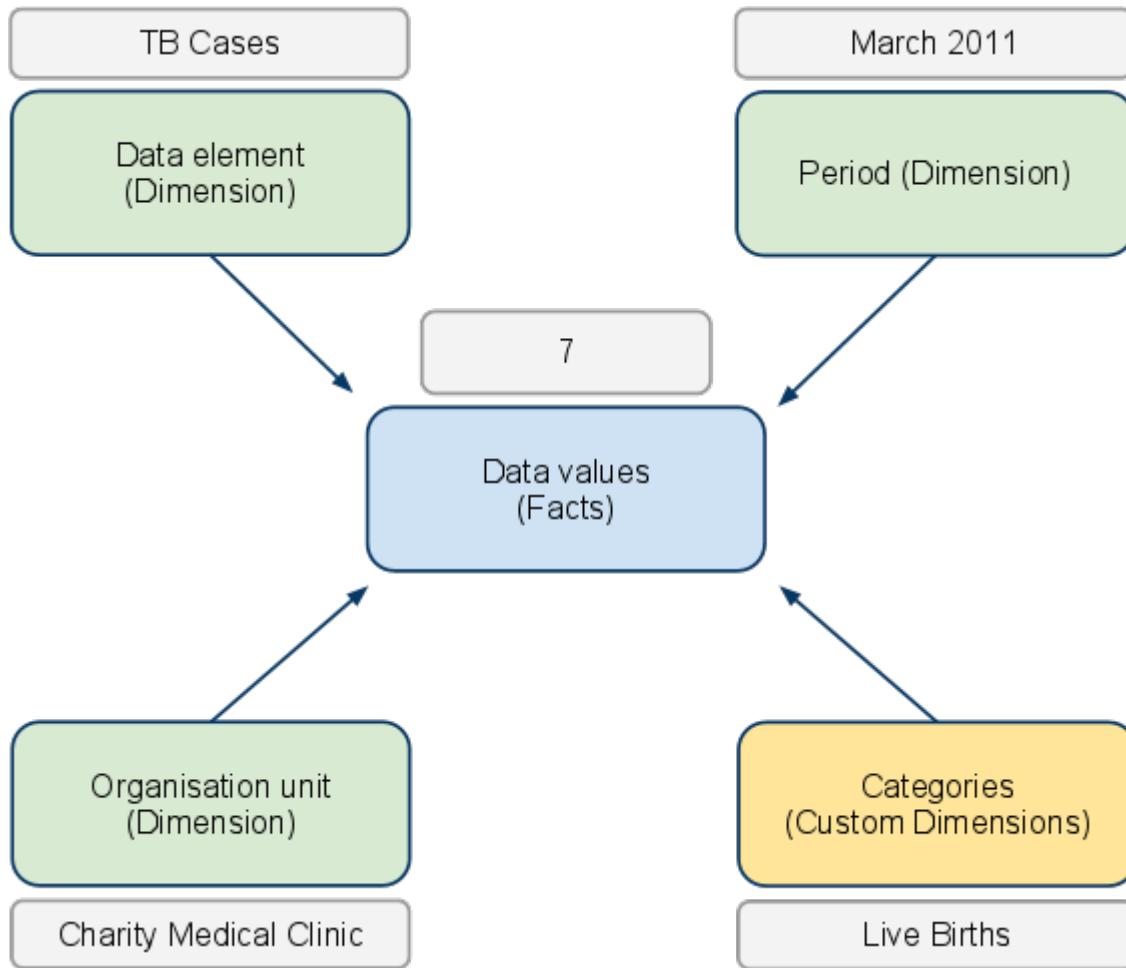
DHIS2 allows for setting up scheduled aggregation tasks which typically will refresh and populate the data mart with aggregated data every night. You can choose between aggregating data for the last 12 months every night, or aggregate data for the last 6 months every night and the last 6 to 12 months every Saturday. The scheduled tasks can be configured under "Scheduling" in "Data administration" module. It is also possible to execute arbitrary data mart tasks under "Data mart" in "Reports" module.

Data storage approach

There are two leading approaches for storing data in a data warehouse, namely the *normalized* and *dimensional* approach. DHIS2 lends a bit from the former but mostly from the latter. In the dimensional approach, the data is partitioned into *dimensions* and *facts*. Facts generally refer to transactional numeric data while dimensions are the reference data that gives context and meaning to the data. The strict rules of this approach make it easy for users to understand the data warehouse structure and provides for good performance since few tables must be combined to produce meaningful analysis, while it, on the other hand, might make the system less flexible and harder to change.

In DHIS2 the facts correspond to the data value object in the data model. The data value captures data as numbers, yes/no or text. The *compulsory dimensions* which give meaning to the facts are the

data element, organisation unit hierarchy and period dimensions. These dimensions are referred to as compulsory since they must be provided for all stored data records. DHIS2 also has a custom dimensional model which makes it possible to represent any kind of dimensionality. This model must be defined prior to data capture. DHIS2 also has a flexible model of groups and group sets which makes it possible to add custom dimensionality to the compulsory dimensions after data capture has taken place. You can read more about dimensionality in DHIS2 in the chapter by the same name.



DHIS2 as a platform

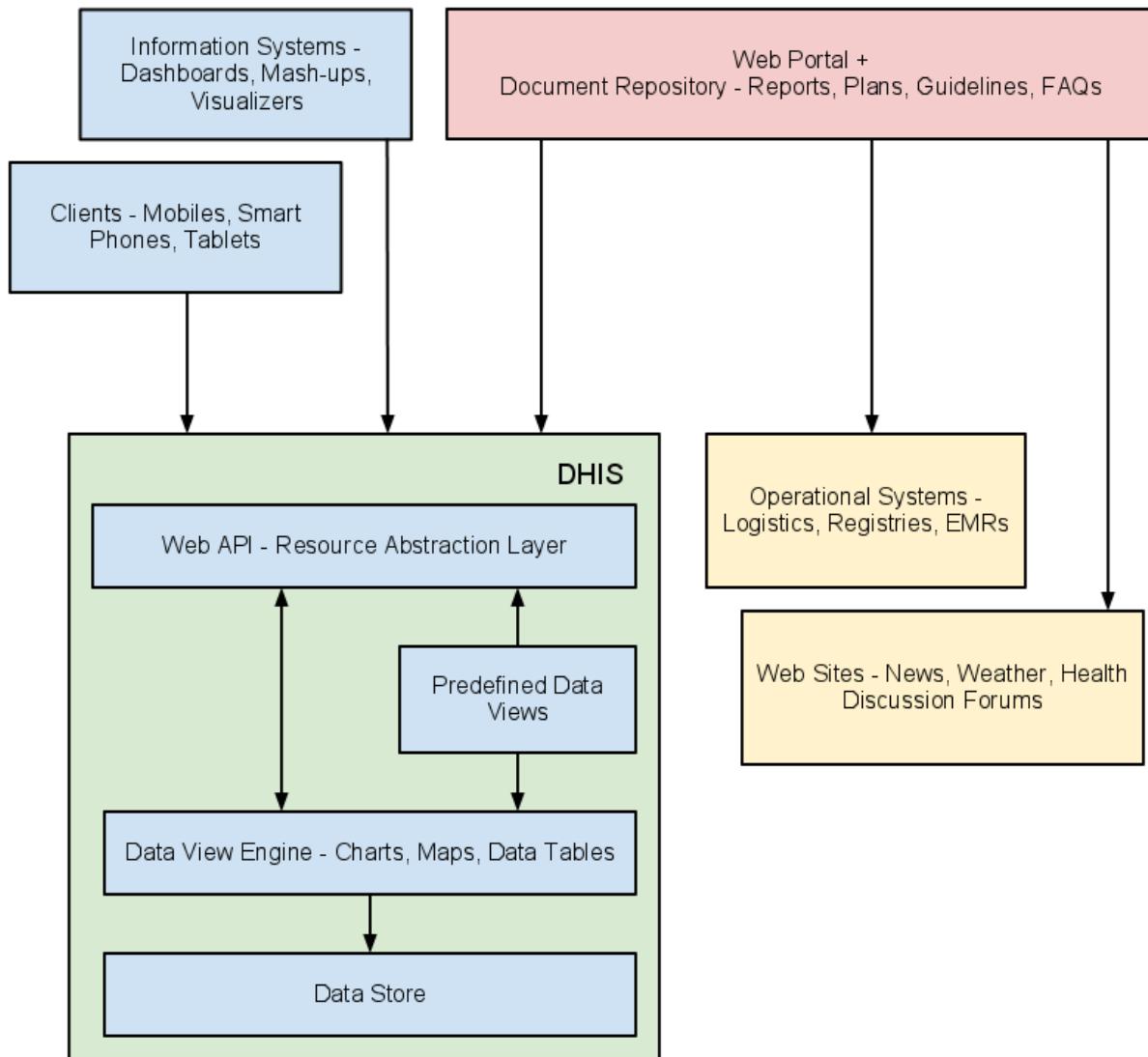
DHIS2 can be perceived as a platform on several levels. First, the application database is designed ground-up with flexibility in mind. Data structures such as data elements, organisation units, forms and user roles can be defined completely freely through the application user interface. This makes it possible for the system to be adapted to a multitude of local contexts and use-cases. We have seen that DHIS2 supports most major requirements for routine data capture and analysis emerging in country implementations. It also makes it possible for DHIS2 to serve as a management system for domains such as logistics, labs and finance.

Second, due to the modular design of DHIS2 it can be extended with additional software modules. These software modules can live side by side with the core modules of DHIS2 and can be integrated into the DHIS2 portal and menu system. This is a powerful feature as it makes it possible to extend the system with extra functionality when needed, typically for country specific requirements as earlier pointed out.

The downside of the software module extensibility is that it puts several constraints on the development process. The developers creating the extra functionality are limited to the DHIS2 technology in terms of programming language and software frameworks, in addition to the constraints put on the design of modules by the DHIS2 portal solution. Also, these modules must be included in the DHIS2 software when the software is built and deployed on the web server, not dynamically during run-time.

In order to overcome these limitations and achieve a looser coupling between the DHIS2 service layer and additional software artifacts, the DHIS2 development team decided to create a Web API. This Web API complies with the rules of the REST architectural style. This implies that:

- The Web API provides a navigable and machine-readable interface to the complete DHIS2 data model. For instance, one can access the full list of data elements, then navigate using the provided hyperlink to a particular data element of interest, then navigate using the provided hyperlink to the list of forms which this data element is part of. E.g. clients will only do state transitions using the hyperlinks which are dynamically embedded in the responses.
- Data is accessed through a uniform interface (URLs) using a well-known protocol. There are no fancy transport formats or protocols involved - just the well-tested, well-understood HTTP protocol which is the main building block of the Web today. This implies that third-party developers can develop software using the DHIS2 data model and data without knowing the DHIS2 specific technology or complying with the DHIS2 design constraints.
- All data including meta-data, reports, maps and charts, known as resources in REST terminology, can be retrieved in most of the popular representation formats of the Web of today, such as HTML, XML, JSON, PDF and PNG. These formats are widely supported in applications and programming languages and give third-party developers a wide range of implementation options.



There are several scenarios where additional software artifacts may connect to the DHIS2 Web API.

Web portals

First, Web portals may be built on top of the Web API. A Web portal in this regard is a web site which functions as a point of access to information from a potential large number of data sources which typically share a common theme. The role of the Web portal is to make such data sources easily accessible in a structured fashion under a common look-and-feel and provide a comprehensive data view for end users.

Aggregate data repository: A Web portal targeted at the health domain may use the DHIS2 as the main source for aggregate data. The portal can connect to the Web API and communicate with relevant resources such as maps, charts, reports, tables and static documents. These data views can dynamically visualize aggregate data based on queries on the organisation unit, indicator or period dimension. The portal can add value to the information accessibility in several ways. It can be structured in a user-friendly way and make data accessible to inexperienced users. It can provide various approaches to the data, including:

- Thematic - grouping indicators by topic. Examples of such topics are immunization, mother care, notifiable diseases and environmental health.
- Geographical - grouping data by provinces. This will enable easy comparison of performance and workload.

Mash-up: The Web portal is not limited to consuming data from a single Web API - it can be connected to any number of APIs and be used to mash up data from auxiliary systems within the health domain. If available the portal might pull in specialized data from logistics systems tracking and managing ARV medicines, from finance systems managing payments to health facilities and from lab systems tracking lab tests for communicable diseases. Data from all of these sources might be presented in a coherent and meaningful way to provide better insight in the situation of the health domain.

Document repository: The Web portal can act as a document repository in itself (also referred to as content management system). Relevant documents such as published reports, survey data, annual operational plans and FAQs might be uploaded and managed in terms of ownership, version control and classification. This makes the portal a central point for document sharing and collaboration. The emergence of high-quality, open source repository/CMS solutions such as Alfresco and Drupal makes this approach more feasible and compelling.

Knowledge management: KM refers to practices for identifying, materializing and distributing insight and experience. In our context it relates to all aspects of information system implementation and use, such as:

- Database design
- Information system usage and how-to
- End-user training guidelines
- Data use, analysis and interpretation

Knowledge and learning within these areas can be materialized in the form of manuals, papers, books, slide sets, videos, system embedded help text, online learning sites, forums, FAQs and more. All of these artifacts might be published and made accessible from the Web portal.

Forum: The portal can provide a forum for hosting discussions between professional users. The subject can range from help for performing basic operations in the health information system to discussions over data analysis and interpretation topics. Such a forum can act as an interactive source for information and evolve naturally into a valuable archive.

Apps

Second, third-party software clients running on devices such as mobile phones, smart phones and tablets may connect to the DHIS2 Web API and read and write to relevant resources. For instance, third-party developers may create a client running on the Android operating system on mobile devices targeted at community health workers who needs to keep track of the people to visit, register vital data for each encounter and receive reminders of due dates for patient care while travelling freely in the community. Such a client application might interact with the patient and activity plan resources exposed by the DHIS2 Web API. The developer will not be dependent on deep insight into the DHIS2 internal implementation, rather just basic skills within HTTP/Web programming and a bit of knowledge of the DHIS2 data model. Understanding the DHIS2 data model is made easier by the navigable nature of the Web API.

Information Systems

Third, information system developers aiming at creating new ways of visualizing and presenting aggregate data can utilize the DHIS2 Web API as the service layer of their system. The effort needed for developing new information systems and maintaining them over time is often largely underestimated. Instead of starting from scratch, a new application can be built on top of the Web API. Developer attention can be directed towards making new, innovative and creative data representations and visualizations, in the form of e.g. dashboards, GIS and charting components.

Integration concepts

DHIS2 is an open platform and its implementers are active contributors to interoperability initiatives, such as openHIE. The DHIS2 application database is designed with flexibility in mind. Data structures such as data elements, organisation units, forms and user roles can be defined completely freely through the application user interface. This makes it possible for the system to be adapted to a multitude of local contexts and use-cases. DHIS2 supports many requirements for routine data capture and analysis emerging in country implementations, both for HMIS scenarios and as a basic data collection and management system in domains such as [logistics, laboratory management and finance](#).

Integration and interoperability

Based on its platform approach, DHIS2 is able to receive and host data from different data sources and share it to other systems and reporting mechanisms. An important distinction of integration concepts is the difference between data integration and systems interoperability:

- When talking about **integration**, we think about the process of making different information systems appear as one, making electronic data available to all relevant users as well as the harmonization of definitions and dimensions so that it is possible to combine the data in useful ways.
- A related concept is **interoperability**, which is one strategy to achieve integration. We consider DHIS2 interoperable with other software applications because of its capability to exchange data. For example, DHIS2 and OpenMRS are interoperable, because they allow to share data definitions and data with each other. Interoperability depends on standards for data formats, interfaces, codes and terminologies. These would ideally be internationally agreed-upon standards, but in practice may also consist of de facto standards (which has wide acceptance and usage but is not necessarily formally balloted in a standards development organisation) and other more local agreements within a particular context.

DHIS2 is often used as an integrated data warehouse, since it contains (aggregate) data from various sources, such as [Mother and Child health, Malaria program, census data, and data on stocks and human resources](#). These data sources share the same platform, DHIS2, and are available all from the same place. These subsystems are thus considered integrated into one system.

Interoperability in addition will integrate data sources from other software applications. For example, if census data is stored in a specialized [civil registry or in a vital events system](#), interoperability between this database and DHIS2 would mean that census data would also be accessible in DHIS2.

Finally, the most basic integration activity (that is not always taken into account in the interoperability discussion) is the possibility to integrate data from existing paper systems or parallel vertical systems into DHIS2. Data will be entered directly into DHIS2 without passing through a different software application. This process is based on creating consistent indicator definitions and can already greatly reduce fragmentation and enhance data analysis through an integrated data repository.

Objectives of integration

In most countries we find many different, **isolated** health information systems, causing many information management challenges. Public Health Information System have seen an explosive and often uncoordinated growth over the last years. Modern information technology makes it less costly to implement ICT4D solutions, which can lead to a high diversity of solutions. A staggering example was the mHealth moratorium declaration of Uganda's MoH in 2012, as a reaction to an avalanche of around [50 mHealth solutions](#) that were implemented within the course of a few years. Most of these solutions were standalone approaches that did not share their data with the national systems and rarely were developed beyond pilot status.

This may lead to the conclusion, that all systems should be connected or that **interoperability is an objective** in itself. However DHIS2 is often employed in contexts, where infrastructure is weak, and where resources to run even basic systems reliably are scarce. Fragmentation is a serious problem in this context, however interoperability approaches can only resolve some of the fragmentation problems - and often interoperability approaches result in an additional layer of complexity.

Example

Complexity of Logistics solutions in Ghana

In the area of Logistics or Supply Chain Management, often a multitude of parallel, overlapping or competing software solutions can be found in a single country. As identified in a [JSI study in 2012](#), eighteen (18!) different software tools were documented as being used within the public health supply chain in Ghana alone.

Systems interoperability therefore seems as one possibility to remove fragmentation and redundancies and give public health officers a concise and balanced picture from available data sources. However the effort of connecting many redundant software solutions would be very high and therefore seems questionable. In a first step, focus should be on **reducing the number of parallel systems** and identifying the most relevant systems, afterwards these relevant systems can be integrated.

On this background, we want to define the major objectives of DHIS2 integration approaches:

- **Calculation of indicators:** Many indicators are based on numerators and denominators from different data sources. Examples include mortality rates, including some mortality data as numerator and population data as denominator, staff coverage and staff workload rates (human resource data, and population and headcount data), immunization rates, and the like. For the calculation, you need both the numerator and denominator data, and they should thus be integrated into a single data warehouse. The more data sources that are integrated, the more indicators can be generated from the central repository.
- **Reduce manual processing** and entering of data: With different data at the same place, there is no need to manually extract and process indicators, or re-enter data into the data warehouse. Especially interoperability between systems of different data types (such as patient registers and aggregate data warehouse) allows software for subsystems to both calculate and share data electronically. This reduces the amount of manual steps involved in data processing, which increases data quality.
- **Reduce redundancies:** Often overlapping and redundant data is being captured by the various parallel systems. For instance will HIV/AIDS related data elements be captured both by both multiple general counselling and testing programs and the specialized HIV/AIDS program. Harmonizing the data collection tools of such programs will reduce the total workload of the end users. This implies that such data sources can be integrated into DHIS2 and harmonized with the existing data elements, which involves both data entry and data analysis requirements.
- Improve **organizational aspects:** If all data can be handled by one unit in the ministry of health, instead of various subsystems maintained by the several health programs, this one unit can be professionalized. With staff which sole responsibility is data management, processing, and analysis, more specialized skills can be developed and the information handling be rationalized.
- Integration of **vertical programs:** The typical government health domain has a lot of existing players and systems. An integrated database containing data from various sources becomes more valuable and useful than fragmented and isolated ones. For instance when analysis of epidemiological data is combined with specialized [HIV/AIDS, TB, financial and human resource](#)

[data, or when immunization is combined with logistics/stock](#) data, it will give a more complete picture of the situation.

DHIS2 can help streamlining and **simplifying system architecture**, following questions such as: What is the objective of the integration effort? Can DHIS2 help reduce the number of systems? Can an DHIS2 integration help provide relevant management information at a lower cost, at a higher speed and with a better data quality than the existing systems? Is DHIS2 the best tool to replace other systems, or is another fit-for-purpose solution that can interoperate with DHIS2 more appropriate? More practical information on defining these objectives can be found in [*STEP 1 of the 6-Step implementation guideline*](#).

Health information exchange

Since there are different use-cases for health information, there are different types of software applications functioning within the health sector. We use the term architecture for health information to describe a plan or overview of the various software applications, their specific uses and data connections. The architecture functions as a plan to coordinate the development and interoperability of various subsystems within the larger health information system. It is advisable to develop a plan that covers all components, including the areas that are currently not running any software, to be able to adequately see the requirements in terms of data sharing. These requirements should then be part of specifications for the software once it is developed or procured.

The [**openhealth information exchange \(openHIE\)**](#) is an interoperable interpretation of this architecture, with an HMIS or DHIS2 often assuming a significant role in it. The openHIE framework has been developed with a clear focus on countries in low resource settings, through the participation of several institutions and development partners, including the Oslo HISP program.

The schematic overview below shows the main elements of the openHIE framework, containing a component layer, an interoperability services layer and external systems. The openHIE component layer covers meta or reference data (Terminology, Clients, Facilities), Personal data (Staff, Patient History) and national health statistics. The purpose is to ensure the availability of the same meta data in all systems that participate in the corresponding data exchange (e.g. indicator definitions, facility naming, coding and classification). In some cases, like the case of the Master Facility Registry, the data may also be used to provide information to the general public through a web portal. While the interoperability layer ensures data brokerage between the different systems, the external systems layer contains several sub-systems, many at point of service level, with often overlapping functional range.

There are different approaches to define an eHealth architecture. In the context of this DHIS2 guideline, we distinguish between approaches based on a 1:1 connection versus approaches based on an n:n connection (many-to-many).

1:1 integration

In many countries a national HMIS is often the first system to be rolled out to a large number of facilities and to manage a large number of data on a monthly or quarterly basis. When countries start to develop their health system architecture further, DHIS2 often will be connected to some other systems. This connection is often done directly through a simple script, which automates a data transfer.

We talk of a 1:1 connection because it is limited to two systems. In the case of an LMIS/HMIS integration, one LMIS will transfer data to DHIS2 as defined in the script. This hands-on approach often represents a first step and is one of the most common use cases on the way to an interoperable openHIE architecture. However, this simplicity also brings along disadvantages: In case a second logistics system would want to transfer data to DHIS2 (e.g. commodity data for a specific disease program), a second script would have to be written, to perform this task. These two scripts would then run independently from another, resulting in two separate 1:1 connections.

n:n integration

A different approach is based on placing a purpose-built software to serve as an **interoperability layer** or BUS approach, managing the data transfer between possibly several systems on either side (n:n). This could be the case if for example you wanted to collect stock level data through different LMIS applications, and then share it to a central warehouse LMIS, the HMIS and some vertical disease programs system. The openHIE reference software to assume this role is the "[OpenHIM](#)", but systems like "[MOTECH](#)" have also been used for this purpose as will be discussed below.

While this approach may result in a higher initial effort, it promises to make further integration project easier, because the interoperability layer is being alimented with definitions and mappings that can be re-used for connecting the next systems.

In practice, there are certain challenges to this approach. It takes a considerable effort of qualified resources to activate APIs and with each new release of any involved system, data flows require re-testing and if necessary adaptations. Also, to be successful these implementation projects typically have to go through a series of [complex steps](#), such as the agreement on an interoperability approach embedded in the national eHealth strategy, the definition of data standards and sustainable maintenance structure, and attaining a stakeholder consensus on data ownership and sharing policies. There can be some long term consequences when data and systems are knitted together - it creates new roles, jobs and tasks which didn't exist before and may not have been planned for (metadata governance, complex system administration, boundary negotiators, etc.).

Example

Grameen DHIS2/CommCare middle layer in Senegal

In [Integration concepts](#), MOTECH serves as technical middle layer between an LMIS for mobile data collection at the health facility level ([CommCare](#)) and DHIS2, allowing to define data mapping, transformation rules and data quality checks. The interface is set-up to transfers data from CommCare Supply to DHIS2 whenever data is saved into a CommCare form at facilities. For each commodity, data on consumption, available stock, losses and stock-out data is transferred from CommCare to DHIS2.

The higher initial investment of the Senegal approach hints towards a more ambitious long-term system architecture, foreseeing that the MOTECH platform may in future serve to accommodate further interoperability task. However we do not see any of the country activities tightly embedded in a text-book eHealth architecture, which would clearly define areas of priority, leading systems for each priority and the relations and resulting APIs between these different components. One may argue that interoperability projects are built on a weak foundation if there is no previous consensus on an architectural master plan. On the other hand it is also valuable to allow system initiatives to organically develop, as long as they are rooted in well-founded country needs.

Architecture, standards and mapping

An important element of an eHealth architecture is the inclusion of **international eHealth standards**. Standards are especially relevant for n:n connections, less so for direct (1:1) connections.

Some standards are on the technical level (e.g. transmission methods), other on the contents side (e.g. WHO 100 core indicators). Gradually aligning national system initiatives to these standards can give countries access to proven solutions, benefitting from medical and technological innovation.

Example

Ghana EPI

The Ghana case illustrates how the WHO EPI reporting requirements serves to define standard data in DHIS2. This standardization at the dataset and terminological level is the basis for the system integration. In the area of DHIS2, work is ongoing with WHO to develop standardized datasets, which could in the future open up new opportunities for interoperability and efficiency gains by offering some consistency of metadata across systems, and also encouraging countries to reuse existing solutions.

At the **language** level, there is a need to be consistent about definitions. If you have two data sources for the same data, they need to be comparable. For example, if you collect malaria data from both standard clinics and from hospitals, this data needs to describe the same thing if they need to be combined for totals and indicators. If a hospital is reporting malaria cases by sex but not age group, and other clinics are reporting by age group but not sex, this data cannot be analysed according to either of these dimensions (even though a total amount of cases can be calculated). There is thus a need to agree on uniform definitions.

In addition to uniform definitions across the various sub-systems, **data exchange standards** must be adopted if data is to be shared electronically. The various software applications would need this to be able to understand each other. DHIS2 is supporting several data formats for import and export, including the most relevant standard ADX. Other software applications are also supporting this, and it allows the sharing of data definitions and aggregate data between them. For DHIS2, this means it supports import of aggregate data that are supplied by other applications, such as [OpenMRS](#) (for patient management) and [iHRIS](#) (for human resources management).

A crucial element of the architecture is how organize data **mapping**. Typically the metadata of different systems does not match exactly. Unless an MoH has been enforcing a consequent data standard policy, different systems will have different codes and labels for a facility. one System may call it *District Hospital - 123*, the other system may refer to it as *Malaria Treatment Centre - 15*. To be able to transfer data, somewhere the information that these two facilities correspond needs to be stored.

In the case of a 1:1 connection, this mapping has to be done and maintained for every connection, in case of an n:n interoperability approach, one side of the definitions can be re-used.

In order to assure that the data can flow smoothly, you need to have clear responsibilities on both sides of the system regarding data maintenance and troubleshooting. For example, there need to be previously defined standard procedures for such activities as adding, renaming, temporarily deactivating or removing a facility on either of the two systems. Changes of database fields that are included in a transferred data record need also to be coordinated in a systematic way.

Aggregate and transactional data

DHIS2 has been expanding its reach into many health systems. Starting from its familiar grounds of aggregate data sets for routine data it has included patient related data and then data in the areas of HR, finance, logistics and laboratory management, moving towards operational or transactional data.

We can differentiate between transactional and aggregate data. A **transactional system** (or operational system from a data warehouse perspective) is a system that collects, stores and modifies detailed level data. This system is typically used on a day-to-day basis for data entry and validation. The design is optimized for fast insert and update performance. DHIS2 can incorporate aggregate data from external data sources, typically aggregated in the space dimension (the organisation unit hierarchy), time dimension (over multiple periods) and for indicator formulas (mathematical expressions including data elements).

When we look at a transactional system, such as a logistics software for the entire supply chain or parts of it, there is one fundamental decision to take: Do you need to track all detailed transactions at

all levels, including such operations as returns, transfer between facilities, barcode reading, batch and expiry management? Or can you get most of your needed decision insight results using aggregate data?

Supply chains may often be well monitored and to some degree, managed, as long as data are reliably available where and when they are needed for operational decisions and for monitoring purposes.. The main indicators *intake, consumption and stock level at the end of period*can be managed without electronic transactions and often suffice to give the big picture of system performance, and may reduce the needs for system investment.

Being realistic about what data need to be collected, how often, and who will be using them is important so you don't create systems that fail due to lack of use or unrealistic expectations about how the data will be used. Digital logistics management systems can work well when they are fully integrated into routine workflows and designed to make the users' jobs easier or more efficient.

Note

The expectation, that more detailed data leads to better logistics management is not always fulfilled. Sometimes the ambitious attempt to regularly collect logistics transaction data results in less data quality, for example because the data recording, which may have to happen on a daily basis instead of a monthly or quarterly basis, is not carried out reliably. On the other hand, if the transactional system is well maintained and monitored, more detailed data can help identify inaccuracies and data quality challenges, reduce wastage (due to expiry or CCE failure), support a recall, manage performance and lead to improvements in supply chain decision making. Analysing detailed data may help to discover root causes of some problems and improve the data quality in the long run.

DHIS2 can assume different roles in interoperability scenarios. A common interoperability scenario is for DHIS2 to receive aggregate data from an operational system, in which case the operational system adds up the transactions before passing it on to DHIS2. However, DHIS2 may to a certain extent also be configured to store detailed transactional data, receiving it from external systems or through direct data entry in DHIS2.

On this basis we try making a comparative overview, comparing aggregate DHIS2 data management with data management of external specialized system. This can serve as a rough orientation, but is not static since both the capabilities of DHIS2 and its interpretation by implementers are broadening with almost each release.

Area	Aggregate DHIS2	External specialized systems
Logistics	Aggregate data, e.g. end-of-month facility stock levels can be send through DHIS2. DHIS2 can produce simple stock level and consumption reports.	Supply chain management support logistics system operations and can track detailed stock movements (Issuing, resupplying, allocating, wastage) and record details such as production batch numbers. SCM systems create forecasting, replenishment and elaborate control reports, allowing for real time monitoring of stock levels, notifications (low stock, workflow management, CCE failure, etc.), supported estimations, and emergency orders.
Finance	Aggregate data, e.g. on total expenditure or cash level can be send through DHIS2. DHIS2 can produce simple finance overview reports, e.g. on remaining budgets.	Finance management systems allow fully traceable recording of financial transactions according to legal requirements, including budgeting, transfers, cancellations, reimbursements etc. Multi-dimensional tagging of transactions allows for analytical reports.
Patient tracking	Disease or program related data are collected by DHIS2, DHIS2 Tracker also allows a simplified longitudinal view on medical records, including patient history and multi-stage clinical pathways.	Specialized hospital management systems can cover and optimize complex workflows between different departments (e.g. reception, payment counter, wards, OPD, IPD, laboratory, imaging, storeroom, finance and HR administration, medical device maintenance, etc.).
Human Resources	DHIS2 collects human resource related indicators, for example planned positions and vacancies per facility.	A specialized HR management system can track detailed status information and changes for a Health Worker (accreditation, promotion, sabbatical, change of position, change of location, additional training, etc.). It comes with pre-designed reports for both operational oversight and planning.

Different DHIS2 integration scenarios

The different objectives described above lead to different integration scenarios. DHIS2 can assume multiple **roles** in a system architecture:

- Data input: data entry (offline, mobile), data import (transactional data, aggregate data)
- Data storage, visualisation and analysis with in-built tools (DWH, reports, GIS)

• Data sharing to external tools (e.g. DVDMT), via web APIs, web apps

• In the following paragraphs we discuss the data input and data sharing approaches, then we present the example of the vertical integration where DHIS2 often assumes all these roles.

The role of DHIS2 to store, visualise and analyse data is discussed separately in the [data warehouse section](#).

Data input

There are several aspects on how DHIS2 deals with data input. On the most basic level, DHIS2 serves to replace or at least mirror paper-based data collection forms, integrating the data electronically. This will result in **manual data entry** activities at facility or at health administration level. The next input option is to **import data**. DHIS2 allows to import data through a user interface, which is a method requiring little technical knowledge, but needs to be executed manually every time data needs to be made available. A detailed description of the import functions can be found in the [DHIS2 user guides](#).

Tip

The manual data entry and import approach require relatively little technical effort. They may also be used temporarily to pilot a data integration approach. This allows user to test indicators and reports, without having to employ dedicated technical resources for the development of automated interoperability functions, either through a 1:1 or an n:n connection.

Data sharing

There are three sharing scenarios, (1) a simple [data export](#), (2) [DHIS2 apps and \(3\) external apps or websites connecting to the DHIS Web API](#). Similar to the import functions described in the data input section, the most accessible way of data sharing is to use the data export functions that are available from the user menu, requiring little technical knowledge.

Due to its modular design DHIS2 can be extended with **additional software modules, which can be downloaded from the DHIS2 App store**. These software modules can live side by side with the core modules of DHIS2 and can be integrated into the DHIS2 portal and menu system. This is a powerful feature as it makes it possible to extend the system with extra functionality when needed, typically for country specific requirements as earlier pointed out.

The downside of the software module extensibility is that it puts several constraints on the development process. The developers creating the extra functionality are limited to the DHIS2 technology in terms of programming language and software frameworks, in addition to the constraints put on the design of modules by the DHIS2 portal solution. Also, these modules must be included in the DHIS2 software when the software is built and deployed on the web server, not dynamically during run-time.

In order to overcome these limitations and achieve a looser coupling between the DHIS2 service layer and additional software artefacts, the DHIS2 development team decided to create a **Web API**. This Web API complies with the rules of the REST architectural style. This implies that:

- The Web API provides a navigable and machine-readable interface to the complete DHIS2 data model. For instance, one can access the full list of data elements, then navigate using the provided hyperlink to a particular data element of interest, then navigate using the provided hyperlink to the list of forms which this data element is part of. E.g. clients will only do state transitions using the hyperlinks which are dynamically embedded in the responses.

- Data is accessed through a uniform interface (URLs) using a well-known protocol. There are no fancy transport formats or protocols involved - just the well-tested, well-understood HTTP protocol which is the main building block of the Web today. This implies that third-party developers can develop software using the DHIS2 data model and data without knowing the DHIS2 specific technology or complying with the DHIS2 design constraints.
- All data including meta-data, reports, maps and charts, known as resources in REST terminology, can be retrieved in most of the popular representation formats of the Web of today, such as HTML, XML, JSON, PDF and PNG. These formats are widely supported in applications and programming languages and gives third-party developers a wide range of implementation options.

This Web API can be accessed by different external information system. The effort needed for developing new information systems and maintaining them over time is often largely underestimated. Instead of starting from scratch, a new application can be built on top of the Web API.

External systems can offer different options for visualizing and presenting DHIS2 data, e.g. in the form of dashboards, GIS and charting components. Web portals targeted at the health domain can use DHIS2 as the main source for aggregate data. The portal can connect to the Web API and communicate with relevant resources such as maps, charts, reports, tables and static documents. These data views can dynamically visualize aggregate data based on queries on the organisation unit, indicator or period dimension. The portal can add value to the information accessibility in several ways. It can be structured in a user-friendly way and make data accessible to inexperienced users. An example for this is the [Tanzania HMIS Web Portal](#).

DHIS2 maturity model

Taking into account all the above elements on system architecture and data types, DHIS2 implementers have several options on how to approach implementations:

- Focus on transactional or aggregate data
- Focus on data integration or systems interoperability

Given the efforts required to implement systems interoperability, many Ministries of Health are going for the pragmatic shortcut of integrating data such as basic stock level data **directly into their existing national DHIS2**. As a rapidly evolving platform, DHIS2 has been adding a lot of functionality over the last years, especially in DHIS2 Tracker. Taking the example of logistics data, the following main functions are currently available:

- Data entry form mirroring the widely used Report and Requisition (R\&R) paper form. Data entry by facilities is possible through the desktop browser or a mobile app, including in offline mode. These electronic forms can be filled by staff based on the paper stock cards, that are normally placed next to the commodity in the store room.
- DHIS2 can then produce reports for central level performance monitoring, giving commodity and program managers an understanding of how the logistics system is functioning.. Depending on how the logistics system operates, these data may also be able to support operational decision-making although a more complete analysis of logistics business processes and users should be conducted first.
- Stock data can be transformed into logistics indicators, that can be put into context with other program indicators, for example cross-referencing number of patients treated with a specific pathology and corresponding drug consumption.

Although each country that we look at in the use cases has their own development path towards system integration, some common learnings can be drawn from their experiences. The maturity model below describes an evolutionary approach to cope with integration and interoperability challenges,

allowing the different stakeholders in a national Health System to grow professional analytics and data usage habits.

The maturity model suggests moving from aggregate data to transactional data and from stand-alone to interoperable systems (using the example of logistics data).

1. DHIS2 is often one of the first systems to cover the health administration and several facility levels of a country. At first core disease indicators are covered (for example corresponding to the 100 WHO Core Health Indicators).
2. In a second phase, different stakeholders seek to complement the disease and service delivery data they are reporting with basic LMIS data. This can be done on an aggregate basis in DHIS2, e.g. by including stock levels and consumption in periodic reports. This will provide high level information on logistics system performance but may or may not provide sufficient insights to support improved logistics system operations.
3. At a more mature stage, there may be a legitimate need for specialized logistics systems, especially when a very detailed transactional view is wanted to have a more granular control, (e.g. returns, transfers between facilities, batch numbers and expiries, etc.). DHIS2 Tracker can offer some event or patient related data management functions, but cannot always achieve the degree of workflow support provided by other, more specialized solutions.
4. In a mature technological and managerial environment, the logistics transactions can be shared to DHIS2 in an aggregate form, moving from a stand-alone to an integrated scenario.

Implementation steps for successful data and system integration

The purpose of this step-by-step DHIS2 Implementation Guide is to provide a methodology for implementers to create and support a DHIS2 integration scenario. The guide is based on the best practices and lessons learned. The guide advocates for a country driven, iterative, and agile approach that begins with collecting user stories and functional requirements. The guide is intended as a framework that can be adapted to the specific context of each country. The content describes specific examples for each step detailing user stories, data specifications, job aids and checklists to guide the use of the reference implementation software. The basic structure, including the 6 steps, are based on the [OpenHIE implementation methodology](#):

Step 1: Identify Stakeholders and Motivations for Improved Facility Data

Step 2: Document Facility Registry Specifications and User Stories

Step 3: Set Up Initial Instance

Step 4: Identify Gaps & Iterative Development via User Testing

Step 5: Scaling the Registry Implementation

Step 6: Provide Ongoing Support

In addition to these steps related to interoperability, it is also interesting to reference back to some of the general DHIS2 implementation experiences and best practises given in the sections on [Recommendations for National HIS Implementations](#) and [Setting Up a New Database](#). A typical DHIS2 implementation approach which is also vital for interoperability projects is a **participatory** approach. This approach stresses to include right from project start [a local team](#) with different skills and background to assume responsibility as soon as possible.

Step 1: Define strategy, stakeholders and data usage objectives

In a first step, the objectives of the integration project will be defined. As with every technology project, there should be a clear **consensus on strategic and functional objectives**. Technological innovation

and feasibility should not be the sole driving force but rather a clearly defined organisational goal. Therefore this step is also intended to answer the question: "Why do we want to connect systems or integrate data from different sources with DHIS2?"

On a practical level, this leads to questions on the data integration approach, such as:

- Do you want to eliminate paper forms or even eliminate data sets that are redundant or not needed anymore?
- Can you integrate the (aggregate) data into DHIS2?
- Can you integrate the detailed (e.g. patient level or transactional) data into DHIS2, using DHIS2 tracker functions?
- If you want to create a data exchange connection between DHIS2 and another system, how do you define ownership and responsibilities?

Activities to answer this question are described below and will lay the groundwork for an DHIS2 interoperability project.

Identify Stakeholders and Motivations

It is in the nature of interoperability projects to have more than one stakeholder. Stakeholders from different areas need to agree on a common system approach, for example the team responsible for the national HMIS (e.g. the M&E department or Planning Department) and the Logistics Department in case of an LMIS implementation. These two main areas often have sub-divisions, e.g. in the logistics area the procurement unit, the warehousing unit, the transport unit. In addition, stakeholders from disease specific programs will have their own regimens and commodity managers. In addition to these local actors, international partners (agencies, donors, INGOs, consultancies) are often also involved in the decision making process.

Therefore it's interesting to look at the main motivations of the stakeholders and how to mitigate risks resulting from potential diverging interests.

- Central MoH Departments such as **M&E &**Planning often are the main stakeholders for a standardisation of indicators and IT Systems
- **Central IT departments** have a general interest over (often locally controlled) technology choices and ownership, hardware and software purchases. They are often dealing with network and hardware issues but lack experience dealing with complex web-based architectures and data exchanges.
- **Specialized disease programs** are often under pressure to deliver very program specific indicators, both for their own management but also responding to donor driven approaches. They may also feel more comfortable controlling their proper IT system to be sure their needs are prioritized.
- **Specialized functional areas** (such as Human Resources, Logistics, Hospital Management) are often in a sandwich position, having to cater to the information needs of several different stakeholders, while trying to achieve operational efficiency with limited resources.

By identifying who is interested to provide or utilize the data, the lead implementers can start to form a project team to inform the design and implementation. One method for characterizing stakeholders involves grouping interested parties by their functional roles. The existing infrastructure and procedures are also important to understanding governance and curation options. Understanding the stakeholders and their corresponding systems is a critical first step.

eHealth System inventory

It is important to get a clear view on the overall IT systems landscape. This can help make sure that interoperability investment is done to strengthen the main systems and that the investments contribute to a **simplification** of the system architecture. For example, if the system inventory shows that there are a lot of redundant functional systems, e.g. more than 10 different logistics systems or modules in a country, the interoperability project should try to contribute to a mid or long-term rationalization of this situation. This could mean to participate in a national consensus finding process to identify the most future-proof solutions, identify national “champions” for each speciality and develop a roadmap for aligning these systems or data and removing underutilized or redundant systems.

Also in this context it is interesting to analyse whether simple indicators can be collected and managed in DHIS2 itself and how this can complement logistics system improvement efforts (as this is later explained in an [LMIS example](#)). Once the stable and sustainable systems have been identified, planning for a data exchange with DHIS2 can start.

Explore Opportunities and Challenges

The motivations driving an implementation can be detailed by the perceived opportunities or challenges that stakeholders face. This might include the desire to share data across systems related to health facilities for supply chain management, monitoring and evaluation, health service delivery and many other systems. User stories and use cases will be documented in depth during Step 2, but a high level vision of motivations to engage with partners is also needed.

Organisation and HR

Clear national policies on data integration, data ownership, routines for data collection, processing, and sharing, should be in place at the start of the project. Often some period of disturbance to the normal data flow will take place during integration, so for many the long-term prospects of a more efficient system will have to be judged against the short-term disturbance. Integration is thus often a stepwise process, where measures need to be taken for this to happen as smoothly as possible.

Example

Ghana CHIM

- **Stakeholder cooperation:** The Ghana Centre for Health Information Management(CHIM) has a clear position towards vertical programs and other partners with proper software initiatives. CHIM establishes DHIS2 as an attractive data collection option, supporting other GHS stakeholders to connect to DHIS2 and to work on a common interoperability strategy, evolving DHIS2 according to stakeholder needs. **This also includes data sharing agreements.**
- **Strong sense of system ownership:** CHIM has a strong determination to build up the necessary know-how inside the CHIM team to configure and maintain the system. The CHIM team consists of Health Information Officers, that combine Public Health and Data Management skills.

Also, having clearly defined **system maintenance and update procedures** can certainly help to manage interoperability.

Example

Ghana CHIM

As an example, in the case of Ghana DHIS2, a clear yearly system update

cycle is in place: Towards the end of each year, new indicators are created and the corresponding paper forms are issued. Staff will receive training and is prepared for data entry. The new form for EPI data was included in this update cycle and EPI staff was prepared for data entry as part of the process. This systematic procedure allows GHS to quickly respond to the needs of stakeholders such as the EPI Programme and accommodate their data and reporting needs with a limited and predictable investment. It puts CHIM in a position to contribute to the rationalization and simplification of the national Health System Architecture, gradually integrating the data management for more **vertical programs**, both on the side of data entry and analytics.

A key principle for HISP is to engage the local team in building the system from the very beginning, with guidance from external experts if needed, and not to delay knowledge transfer towards the end of the implementation. Ownership comes first of all from building the system and owning every step of this process.

Step 2: Document Specifications and Requirements

- Collect existing metadata
- Document data specifications
- Document user stories

Step 3: Carry Out Specifications and Identify Gaps

- Implement the specifications
- Identify and prioritize incomplete user stories

Step 4: Iteration and User Testing

- Agile and iterative development
- User testing
- Collect, reconcile and upload data

Step 5: Scale-Up

- Confirm user roles and responsibilities
- User training
- Critical integrations

Step 6: Ongoing Support

While during the implementation phase a temporary support structure should be available, afterwards a permanent support structure needs to be set-up. The main challenge is to have clear responsibilities. In an ideal situation, we are dealing with two stable systems that each have already their own clearly defined support structure.

However in reality some recurring challenges may have to be dealt with: Many Public Health System are undergoing dynamic developments, leading to changes in data collection needs or calculation of indicators.

Interoperability tends to be a tedious technical and organisational challenge. All of the three described initiatives have consumed a considerable effort of qualified **resources** to activate APIs. In addition, with each new release of any involved system, data flows require re-testing and if necessary adaptations. To be successful these implementation projects typically have to go through a series of complex steps, such as the agreement on an interoperability approach embedded in the national eHealth strategy, the definition of data standards and sustainable maintenance structure, and attaining a stakeholder consensus on data ownership and sharing policies. There can be some long term consequences when data and systems are knitted together - it creates **new roles, tasks and categories of labour** which need to be planned for (metadata governance, complex system administration, boundary negotiators, etc.). A solution could be to discuss the new responsibilities beforehand, assigning them to job descriptions, teams and specific positions.

Metadata responsibility

Another important area is that of **metadata governance**, particularly in the scenarios of secondary use of data. In a stand-alone set-up, metadata, such as facility or commodity codes can be managed without much consideration of other stakeholder's needs. But in an interoperability environment, metadata changes will have effects outside of the individual system. Metadata governance can be highly formalised through registries or more manual through human processes.

In order to determine the appropriate approach, is it useful to estimate the expected *metadata maintenance effort* and the consequences of unsynchronized metadata across different systems. In the case of the LMIS/DHIS2 integrations, there are potentially thousands of facility identifiers that could go out of sync. However normally, facility identifiers do not change often since the physical infrastructure of most public health system is relatively constant. As to the commodities, although regimes and priority drugs may change over time, the number of datasets is relatively small: The commodity list of a program often contains less than 20 products. Therefore often it can be practical to update a commodity manually, and not invest into an interoperability solutions such as an automated metadata synchronization.

Specific integration and interoperability use cases

DHIS2 has been expanding its reach into many health systems. Starting from its familiar grounds of aggregate data sets for routine data it has included patient related data and then data in the areas of HR, finance, logistics and laboratory management. This is in line with the development of DHIS2 in many country settings, where implementers are pushing the use beyond its originally intended scope.

This is also reflected in the overall system architecture. Since the expanding functionality of DHIS2 reduces the urgency to introduce or maintain other specialized systems, the number of potential data interfaces decreases. This **reduced complexity** in system architecture is certainly a benefit for a Health System with limited resources.

For several years now, DHIS2 has grown its data management activities organically, allowing the actual usage to lead to sometimes unforeseen solutions. However, there are also limits to where leveraging DHIS2 seems useful. In the following sections, special systems will be described.

Logistics Management

a) Introduction

Logistics Management Systems (**LMIS**) or Supply Chain Management Systems (**SCM**) serve to replace paper systems to increase standardization, transparency, timeliness of procurement, efficiency, safety, cost-effectiveness, and to reduce waste. National SCMS/LMIS can cover such functions as commodity planning, budgeting, procurement, storage, distribution and replenishment of essential drugs and consumables.

b) Implementing LMIS in DHIS2

Supply chains can often be well controlled with aggregate data only, as long as data is provided reliably from all relevant levels and followed up upon. The main indicators intake, consumption and stock level at the end of period can be managed without electronic transactions and often suffice to give the big picture, reducing the needs for system investment. As a rapidly evolving platform, DHIS2 has been adding a lot of functionality over the last years, especially in DHIS2 Tracker. The following main functions are currently available:

- Data entry form mirroring the widely used Report and Requisition (R\&R) paper form. Data entry by facilities is possible through the desktop browser or a mobile app, including in offline mode. In online mode the form can calculate requisition proposals, offering the facility manager to modify the request and comment on it. These electronic forms can be filled by staff based on the paper stock cards, that are normally placed next to the commodity in the store room.
- DHIS2 can then produce reports for central decision making, giving commodity and program managers the possibility to accept or modify delivery suggestions.
- Stock data can be transformed into logistics indicators, that can be put into context with other program indicators, for example cross-referencing number of patients treated with a specific pathology and corresponding drug consumption.

c) Interoperability Options

LMIS is an area where a multitude of parallel, overlapping or competing software solutions can be found in a single country. As identified in a JSI study in 2012(Ghana Ministry of Health, July 2013: Landscape Analysis of Supply Chain Management Tools in Use), eighteen (18!) different software tools were documented as being in use within the public health supply chain in Ghana alone.

Although a basic LMIS configuration based on aggregate data can take you very far, in some cases a transactional LMIS is necessary if you need to track such detailed operations as returns, transfer between facilities, barcode reading, batch and expiry management. Also some specialized HQ functions such as creating forecasting, replenishment and elaborate control reports are often done in specialized tools.

DHIS2 has integrated aggregate data from external systems such as openLMIS and CommCare through automated data interfaces. As a result, stock data is available in shared dashboards, displaying health service and stock data next to each other.

Data Quality

This chapter discusses various aspects related to data quality.

Measuring data quality

Is the data complete? Is it collected on time? Is it correct? These are questions that need to be asked when analysing data. Poor data quality can take many shapes; not just incorrect figures, but a lack of completeness, or the data being too old (for meaningful use).

Reasons for poor data quality

There are many potential reasons for poor quality data, including:

- Excessive amounts collected; too much data to be collected leads to less time to do it, and “shortcuts” to finish reporting
- Many manual steps; moving figures, summing up, etc. between different paper forms
- Unclear definitions; wrong interpretation of the fields to be filled out
- Lack of use of information: no incentive to improve quality
- Fragmentation of information systems; can lead to duplication of reporting

Improving data quality

Improving data quality is a long-term task, and many of the measures are organizational in nature. However, data quality should be an issue from the start of any implementation process, and there are some things that can be addressed at once, such as checks in DHIS2. Some important data quality improvement measures are:

- Changes in data collection forms, harmonization of forms
- Promote information use at local level, where data is collected
- Develop routines on checking data quality
- Include data quality in training
- Implement data quality checks in DHIS2

Using DHIS2 to improve data quality

DHIS2 has several features that can help the work of improving data quality; validation during data entry to make sure data is captured in the right format and within a reasonable range, user-defined validation rules based on mathematical relationships between the data being captured (e.g. subtotals vs totals), outlier analysis functions, as well as reports on data coverage and completeness. More indirectly, several of the DHIS2 design principles contribute to improving data quality, such as the idea of harmonising data into one integrated data warehouse, supporting local level access to data and analysis tools, and by offering a wide range of tools for data analysis and dissemination. With more structured and harmonised data collection processes and with strengthened information use at all levels, the quality of data will improve. Here is an overview of the functionality more directly targeting data quality:

Data input validation

The most basic type of data quality check in DHIS2 is to make sure that the data being captured is in the correct format. The DHIS2 will give the users a message that the value entered is not in the correct format and will not save the value until it has been changed to an accepted value. E.g. text

cannot be entered in a numeric field. The different types of data values supported in DHIS2 are explained in the user manual in the chapter on data elements.

Min and max ranges

To stop typing mistakes during data entry (e.g typing '1000' instead of '100') the DHIS2 checks that the value being entered is within a reasonable range. This range is based on the previously collected data by the same health facility for the same data element, and consists of a minimum and a maximum value. As soon as a user enters a value outside the user will be alerted that the value is not accepted. In order to calculate the reasonable ranges the system needs at least six months (periods) of data.

Validation rules

A validation rule is based on an expression which defines a relationship between a number of data elements. The expression has a left side and a right side and an operator which defines whether the former must be less than, equal to or greater than the latter. The expression forms a condition which should assert that certain logical criteria are met. For instance, a validation rule could assert that the total number of vaccines given to infants is less than or equal to the total number of infants. The left and right sides must return numeric values.

The validation rules can be defined through the user interface and later be run to check the existing data. When running validation rules the user can specify the organisation units and periods to check data for, as running a check on all existing data will take a long time and might not be relevant either. When the checks have completed a report will be presented to the user with validation violations explaining which data values need to be corrected.

The validation rules checks are also built into the data entry process so that when the user has completed a form the rules can be run to check the data in that form only, before closing the form.

Outlier analysis

The standard deviation based outlier analysis provides a mechanism for revealing values that are numerically distant from the rest of the data. Outliers can occur by chance, but they often indicate a measurement error or a heavy-tailed distribution (leading to very high numbers). In the former case one wishes to discard them while in the latter case one should be cautious in using tools or interpretations that assume a normal distribution. The analysis is based on the standard normal distribution.

Completeness and timeliness reports

Completeness reports will show how many data sets (forms) have been submitted by organisation unit and period. You can use one of three different methods to calculate completeness; 1) based on completeness button in data entry, 2) based on a set of defined compulsory data elements, or 3) based on the total registered data values for a data set.

The completeness reports will also show which organisation units in an area are reporting on time, and the percentage of timely reporting facilities in a given area. The timeliness calculation is based on a system setting called Days after period end to qualify for timely data submission.

Organisation Units

In DHIS2 the location of the data, the geographical context, is represented as organisational units. Organisational units can be either a health facility or department/sub-unit providing services or an administrative unit representing a geographical area (e.g. a health district).

Organisation units are located within a hierarchy, also referred to as a tree. The hierarchy will reflect the health administrative structure and its levels. Typical levels in such a hierarchy are the national, province, district and facility levels. In DHIS2 there is a single organisational hierarchy so the way this is defined and mapped to the reality needs careful consideration. Which geographical areas and levels are defined in the main organisational hierarchy will have major impact on the usability and performance of the application. Additionally, there are ways of addressing alternative hierarchies and levels as explained in the section called Organisation unit groups and group sets further down.

Organisation unit hierarchy design

The process of designing a sensible organisation unit hierarchy has many aspects:

- *Include all reporting health facilities:* All health facilities which contribute to the national data collection should be included in the system. Facilities of all kinds of ownership should be incorporated, including private, public, NGO and faith-oriented facilities. Often private facilities constitute half of the total number of facilities in a country and have policies for data reporting imposed on them, which means that incorporating data from such facilities are necessary to get realistic, national aggregate figures.
- *Emphasize the health administrative hierarchy:* A country typically has multiple administrative hierarchies which are often not well coordinated nor harmonized. When considering what to emphasize when designing the DHIS2 database one should keep in mind what areas are most interesting and will be most frequently requested for data analysis. DHIS2 is primarily managing health data and performing analysis based on the health administrative structure. This implies that even if adjustments might be made to cater for areas such as finance and local government, the point of departure for the DHIS2 organisation unit hierarchy should be the health administrative areas.
- *Limit the number of organisation unit hierarchy levels:* To cater for analysis requirements coming from various organisational bodies such as local government and the treasury, it is tempting to include all of these areas as organisation units in the DHIS2 database. However, due to performance considerations one should try to limit the organisation unit hierarchy levels to the smallest possible number. The hierarchy is used as the basis for aggregation of data to be presented in any of the reporting tools, so when producing aggregate data for the higher levels, the DHIS2 application must search for and add together data registered for all organisation units located further down the hierarchy. Increasing the number of organisation units will hence negatively impact the performance of the application and an excessively large number might become a significant problem in that regard.

In addition, a central part in most of the analysis tools in DHIS2 is based around dynamically selecting the “parent” organisation unit of those which are intended to be included. For instance, one would want to select a province and have the districts belonging to that province included in the report. If the district level is the most interesting level from an analysis point of view and several hierarchy levels exist between this and the province level, this kind of report will be rendered unusable. When building up the hierarchy, one should focus on the levels that will be used frequently in reports and data analysis and leave out levels that are rarely or never used as this will have an impact on both the performance and usability of the application.

- *Avoid one-to-one relationships:* Another guiding principle for designing the hierarchy is to avoid connecting levels that have near one-to-one parent-child ratios, meaning that for instance a

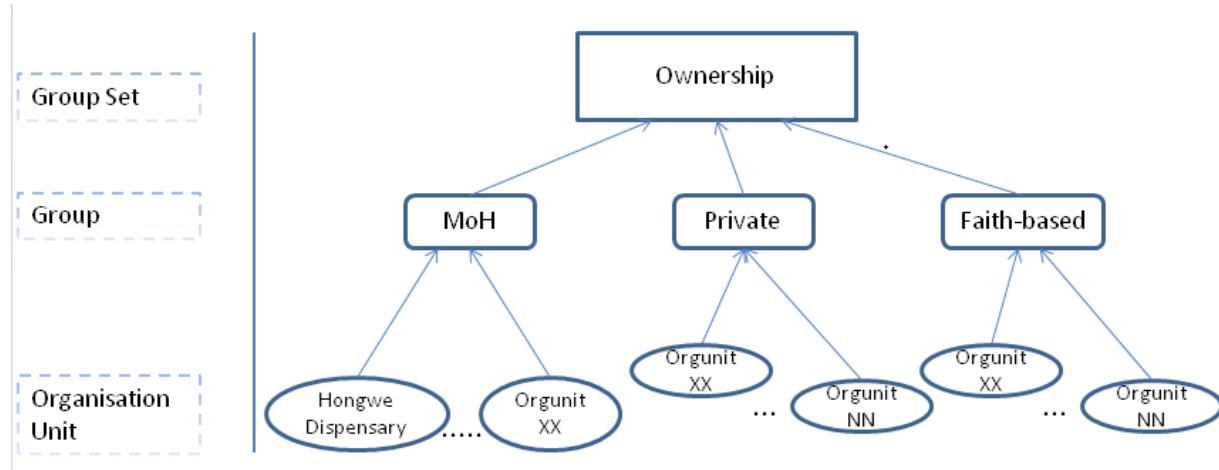
district (parent) should have on average more than one local council (child) at the level below before it make sense to add a local council level to the hierarchy. Parent-child ratios from 1:4 or more are much more useful for data analysis purposes as one can start to look at e.g. how a district's data is distributed in the different sub-areas and how these vary. Such drill-down exercises are not very useful when the level below has the same target population and the same serving health facilities as the parent.

Skipping geographical levels when mapping the reality to the DHIS2 organisation unit hierarchy can be difficult and can easily lead to resistance among certain stakeholders, but one should have in mind that there are actually ways of producing reports based on geographical levels that are not part of the organisational hierarchy in DHIS2, as will be explained in the next section.

Organisation unit groups and group sets

In DHIS2, organisation units can be grouped in organisation unit groups, and these groups can be further organised into group sets. Together they can mimic an alternative organisational hierarchy which can be used when creating reports and other data output. In addition to representing alternative geographical locations not part of the main hierarchy, these groups are useful for assigning classification schemes to health facilities, e.g. based on the type or ownership of the facilities. Any number of group sets and groups can be defined in the application through the user interface, and all these are defined locally for each DHIS2 database.

An example illustrates this best: Typically one would want to provide analysis based on the ownership of the facilities. In that case one would create a group for each ownership type, for instance "MoH", "Private" and "NGO". All facilities in the database must then be classified and assigned to one and only one of these three groups. Next one would create a group set called "Ownership" to which the three groups above are assigned, as illustrated in the figure below.



In a similar way one can create a group set for an additional administrative level, e.g. local councils. All local councils must be defined as organisation unit groups and then assigned to a group set called "Local Council". The final step is then to assign all health facilities to one and only one of the local council groups. This enables the DHIS2 to produce aggregate reports by each local council (adding together data for all assigned health facilities) without having to include the local council level in the main organisational hierarchy. The same approach can be followed for any additional administrative or geographical level that is needed, with one group set per additional level. Before going ahead and designing this in DHIS2, a mapping between the areas of the additional geographical level and the health facilities in each area is needed.

A key property of the group set concept in DHIS2 to understand is *exclusivity*, which implies that an organisation unit can be member of exactly one of the groups in a group set. A violation of this rule

would lead to duplication of data when aggregating health facility data by the different groups, as a facility assigned to two groups in the same group set would be counted twice.

With this structure in place, DHIS2 can provide aggregated data for each of the organisation unit ownership types through the “Organisation unit group set report” in “Reporting” module or through the Excel pivot table third-party tool. For instance one can view and compare utilisation rates aggregated by the different types of ownership (e.g. MoH, Private, NGO). In addition, DHIS2 can provide statistics of the distribution of facilities in “Organisation unit distribution report” in “Reporting” module. For instance one can view how many facilities exist under any given organisation unit in the hierarchy for each of the various ownership types. In the GIS module, given that health facility coordinates have been registered in the system, one can view the locations of the different types of health facilities (with different symbols for each type), and also combine this information with another map layer showing indicators e.g. by district.

Data Elements and Custom Dimensions

This chapter first discusses an important building block of the system: the data element. Second it discusses the category model and how it can be used to achieve highly customized meta-data structure for storage of data.

Data elements

The data element is together with the organisation unit the most important building block of a DHIS2 database. It represents the *what* dimension and explains what is being collected or analysed. In some contexts this is referred to an indicator, however in DHIS2 this meta-data element of data collection and analysis is referred to as a data element. The data element often represents a count of some event and its name describes what is being counted, e.g. "BCG doses given" or "Malaria cases". When data is collected, validated, analysed or presented it is the data elements or expressions built with data elements that describe what phenomenon, event or case the data is registered for. Hence the data elements become important for all aspects of the system and decide not only how data is collected, but more importantly how the data is represented in the database and how data can be analysed and presented.

An important principle behind designing data elements is to think of data elements as a self-contained description of an phenomenon or event and not as a field in a data entry form. Each data element lives on its own in the database, completely detached and independent from the collection form. It is important to consider that data elements are used directly in reports, charts and other tools for data analysis, in which the context in any given data entry form is not accessible nor relevant. In other words, it must be possible to clearly identify what event a data element represents by only looking at its name. Based on this one can derive a rule of thumb saying that the name of the data element must be able to stand on its own and describe the data value also outside the context of its collection form.

For instance, a data element called "Malaria" might be concise when seen in a data entry form capturing immunization data, in a form capturing vaccination stocks as well as in a form for out-patient data. When viewed in a report, however, outside the context of the data entry form, it is impossible to decide what event this data element represents. If the data element had been called "Malaria cases", "Malaria stock doses received" or "Malaria doses given" it would have been clear from a user perspective what the report is trying to express. In this case we are dealing with three different data elements with completely different semantics.

Categories and custom dimensions

Certain requirements for data capture necessitate a fine-grained breakdown of the dimension describing the event being counted. For instance one would want to collect the number of "Malaria cases" broken down on gender and age groups, such as "female", "male" and " < 5 years" and " > 5 years". What characterizes this is that the breakdown is typically repeated for a number of "base" data elements: For instance one would like to reuse this break-down for other data elements such as "TB" and "HIV". In order to make the meta-data more dynamic, reusable and suitable for analysis it makes sense to define the mentioned diseases as data elements and create a separate model for the breakdown attributes. This can be achieved by using the category model, which is described in the following.

The category model has three main elements which is best described using the above example:

1. The category option, which corresponds to "female", "male" and " < 5 years" and " > 5 years".
2. The category, which corresponds to "gender" and "age group".
3. The category combination, which should in the above example be named "gender and age group" and be assigned both categories mentioned above.

This category model is in fact self-standing but is in DHIS2 loosely coupled to the data element. Loosely coupled in this regard means that there is an association between data element and category combination, but this association may be changed at any time without losing any data. It is however not recommended to change this often since it makes the database less valuable in general since it reduces the continuity of the data. Note that there is no hard limit on the number of category options in a category or number of categories in a category combination, however there is a natural limit to where the structure becomes messy and unwieldy.

A pair of data element and category combination can now be used to represent any level of breakdown. It is important to understand that what is actually happening is that a number of custom dimensions are assigned to the data. Just like the data element represents a mandatory dimension to the data values, the categories add custom dimensions to it. In the above example we can now, through the DHIS2 output tools, perform analysis based on both “gender” and “age group” for those data elements, in the same way as one can perform analysis based on data elements, organisation units and periods.

This category model can be utilized both in data entry form designs and in analysis and tabular reports. For analysis purposes, DHIS2 will automatically produce sub-totals and totals for each data element associated with a category combination. The rule for this calculation is that all category options should sum up to a meaningful total. The above example shows such a meaningful total since when summarizing “Malaria cases” captured for “female < 5 years”, “male < 5 years”, “female > 5 years” and “male > 5 years” one will get the total number of “Malaria cases”.

For data capture purposes, DHIS2 can automatically generate tabular data entry forms where the data elements are represented as rows and the category option combinations are represented as columns. This will in many situations lead to compelling forms with a minimal effort. It is necessary to note that this however represents a dilemma as these two concerns are sometimes not compatible. For instance one might want to quickly create data entry forms by using categories which do not adhere to the rule of a meaningful total. We do however consider this a better alternative than maintaining two independent and separate models for data entry and data analysis.

An important point about the category model is that data values are persisted and associated with a category option combination. This implies that adding or removing categories from a category combination renders these combinations invalid and a low-level database operation must be done to correct it. It is hence recommended to thoughtfully consider which breakdowns are required and to not change them too often.

Data element groups

Common properties of data elements can be modelled through what is called data element groups. The groups are completely flexible in the sense that both their names and their memberships are defined by the user. Groups are useful both for browsing and presenting related data, and can also be used to aggregate values captured for data elements in the group. Groups are loosely coupled to data elements and not tied directly to the data values which means they can be modified and added at any point in time without interfering with the low-level data.

Data Sets and Forms

This chapter discusses data sets and forms, what types of forms are available and describes best practises for the process of moving from paper based to electronic forms.

What is a data set?

All data entry in DHIS2 is organised through the use of data sets. A data set is a collection of data elements grouped together for data collection, and in the case of distributed installs they also define chunks of data for export and import between instances of DHIS2 (e.g. from a district office local installation to a national server). Data sets are not linked directly to the data values, only through their data elements and frequencies, and as such a data set can be modified, deleted or added at any point in time without affecting the raw data already captured in the system, but such changes will of course affect how new data will be collected.

A data set has a period type which controls the data collection frequency, which can be daily, weekly, monthly, quarterly, six-monthly, or yearly. Both the data elements to include in the data set and the period type is defined by the user, together with a name, short name, and code. If calculated fields are needed in the collection form (and not only in the reports), then indicators can be assigned to the data set as well, but these can only be used in custom forms (see further down).

In order to use a data set to collect data for a specific organisation unit the user must assign the organisation unit to the data set. This mechanism controls which organisation units can use which data sets, and at the same time defines the target values for data completeness (e.g. how many health facilities in a district are expected to submit the RCH data set every month).

A data element can belong to multiple data sets, but this requires careful thinking as it may lead to overlapping and inconstant data being collected if e.g. the data sets are given different frequencies and are used by the same organisation units.

What is a data entry form?

Once you have assigned a data set to an organisation unit that data set will be made available in Data Entry (under Services) for the organisation units you have assigned it to and for the valid periods according to the data set's period type. A default data entry form will then be shown, which is simply a list of the data elements belonging to the data set together with a column for inputting the values. If your data set contains data elements with categories such as age groups or gender, then additional columns will be automatically generated in the default form based on the categories. In addition to the default list-based data entry form there are two more alternatives, the section-based form and the custom form.

Types of data entry forms

DHIS2 currently features three different types of forms which are described in the following.

Default forms

A default data entry form is simply a list of the data elements belonging to the data set together with a column for inputting the values. If your data set contains data elements with a non-default category combination, such as age groups or gender then additional columns will be automatically generated in the default form based on the different options/dimensions. If you use more than one category combination in a data set you will get one table per category combination in the default form, with different column headings for the options.

Section forms

Section forms allow for a bit more flexibility when it comes to using tabular forms and are quick and simple to design. Often your data entry form will need multiple tables with subheadings, and sometimes you need to disable (grey out) a few fields in the table (e.g. some categories do not apply to all data elements), both of these functions are supported in section forms. After defining a data set you can define its sections with subsets of data elements, a heading and possible grey fields in the section's table. The order of sections in a data set can also be defined. In Data Entry you can now start using the Section form (which should appear automatically when sections are available for the selected data set). Most tabular data entry forms should be possible to do with sections forms. Utilizing the section or default forms makes life easy as there is no need to maintain a fixed form design which includes references to data elements. If these two types of forms are not meeting your requirements then the third option is the completely flexible, although more time-consuming, custom data entry forms.

Custom Forms

When the form you want to design is too complicated for the default or section forms then your last option is to use a custom form. This takes more time, but gives you full flexibility in terms of the design. In DHIS2 there is a built in HTML editor (CK Editor) in the form designer which allows you to either design the form in the GUI or paste in your html directly (using the "source" window in the editor). In the custom form you can insert static text or data fields (linked to data elements)

- category option combination) in any position on the form and you have complete freedom to design the layout of the form. Once a custom form has been added to a data set it will be available in Data Entry and used automatically.

When using a custom form it is possible to use calculated fields to display e.g. running totals or other calculations based on the data captured in the form. This can e.g. be useful when dealing with stock or logistics forms that need item balance, items needed for next period etc. In order to do so, the user must first define the calculated expressions as indicators and then assign these indicators to the data set in question. In the custom form designer the user can then assign indicators to the form the same way data elements are assigned. The limitation to the calculated expression is that all the data elements used in the expression must be available in the same data set since the calculations are done on the fly inside the form, and are not based on data values already stored in the database.

From paper to electronic form - Lessons learned

When introducing an electronic health information system the system being replaced is often paper based reporting. The process of migrating to electronic data capture and analysis has some challenges. The following sections suggest best practises on how to overcome these.

Identify self-contained data elements

Typically the design of a DHIS2 data set is based on some requirements from a paper form that is already in use. The logic of paper forms are not the same as the data element and data set model of DHIS, e.g. often a field in a tabular paper form is described both by column headings and text on each row, and sometimes also with some introductory table heading that provides more context. In the database this is captured for one atomic data element with no reference to a position in a visual table format so it is important to make sure the data element, with the optional data element categories, captures the full meaning of each individual field in the paper form.

Leave calculations and repetitions to the computer - capture raw data only

Another important thing to have in mind while designing data sets is that the data set and the corresponding data entry form (which is a data set with layout) is a data collection tool and not a report or analysis tool. There are other far more sophisticated tools for data output and reporting in DHIS2

than the data entry forms. Paper forms are often designed with both data collection and reporting in mind and therefore you might see things such as cumulative values (in addition to the monthly values), repetition of annual data (the same population data reported every month) or even indicator values such as coverage rates in the same form as the monthly raw data. When you store the raw data in the DHIS2 database every month and have all the processing power you need within the computerised tool, there is no need (in fact it would be wrong and most likely cause inconsistency) to register manually calculated values such as the ones mentioned above. You only want to capture the raw data in your data sets/forms and leave the calculations to the computer, and presentation of such values to the reporting tools in DHIS. Through the functionality of data set reports all tabular section forms will automatically get extra columns at the far right providing subtotal and total values for each row (data element).

Indicators

This chapter covers the following topics:

- What is an indicator
- Purposes of indicators
- Indicator-driven data collection
- Managing indicators in DHIS2

The following describes these topics in greater detail.

What is an indicator?

In DHIS2, the indicator is a core element of data analysis. An indicator is a calculated formula based on a combination of data elements, category options, possibly constants and a factor. There are two forms of indicators, those with a denominator and those which do not have a denominator. Calculated totals, which may be composed of multiple data elements do not have denominators. Coverage indicators (ratios, percentages, etc) are composed of two formulas of data elements, one representing the numerator and another representing the denominator.

Indicators are thus made up of formulas of data elements and other components and are always multiplied by a factor (e.g. 1, 100, 100, 100 000). The factor is essentially a number which is multiplied by the result of the numerator divided by denominator. As a concrete example, the indicator "BCG coverage <1 year" is defined by a formula with a factor 100 (in order to obtain a percentage), a numerator ("BCG doses given to children under 1 year") and a denominator ("Target population under 1 year"). The indicator "DPT1 to DPT3 drop out rate" is a formula of $100 \% \times (\text{"DPT1 doses given"} - \text{"DPT3 doses given"}) / (\text{"DPT1 doses given"})$.

Indicator examples

Indicator	Formula	Numerator	Denominator	Factor
Fully immunized <1 year coverage	Fully immunized/ Population < 1 year x 100	Fully immunized	Population < 1	100 (Percentage)
Maternal Mortality Rate	Maternal deaths/ Live births x 100 000	Maternal deaths	Live births	100 000 (MMR is measured per 100 000)
Cumulative number of people Enrolled in Care	Cumulative number of people Enrolled in Care x 1	Cumulative number Enrolled in Care (Male, Age<18) +Cumulative number Enrolled in Care (Male, Age18+) +Cumulative number Enrolled in Care (Female, Age<18) +Cumulative number Enrolled in Care (Female, Age18+)	None	1

Purpose of indicators

Indicators which are defined with both numerators and denominators are typically more useful for analysis. Because they are proportions, they are comparable across time and space, which is very important since units of analysis and comparison, such as districts, vary in size and change over time. A district with population of 1000 people may have fewer cases of a given disease than a district with a population of 10,000. However, the incidence values of a given disease will be comparable between the two districts because of the use of the respective populations for each district.

Indicators thus allow comparison, and are the prime tool for data analysis. DHIS2 should provide relevant indicators for analysis for all health programs, not just the raw data. Most report modules in DHIS2 support both data elements and indicators and you can also combine these in custom reports.

Indicator-driven data collection

Since indicators are more suited for analysis compared to data elements, the calculation of indicators should be the main driving force for collection of data. A usual situation is that much data is collected but never used in any indicator, which significantly reduces the usability of the data. Either the captured data elements should be included in indicators used for management or they should probably not be collected at all.

For implementation purposes, a list of indicators used for management should be defined and implemented in DHIS2. For analysis, training should focus on the use of indicators and why these are better suited than data elements for this purpose.

Managing indicators

Indicators can be added, deleted, or modified at any time in DHIS2 without affecting the data. Indicators are not stored as values in DHIS2, but as formulas, which are calculated whenever the user needs them. Thus a change in the formulas will only lead to different data elements being called for when using the indicator for analysis, without any changes to the underlying data values taking place.

For information how to manage indicators, please refer to the chapter on indicators in the DHIS2 user documentation.

Users and user roles

About user management

Multiple users can access DHIS2 simultaneously and each user can have different authorities. You can fine-tune these authorities so that certain users can only enter data, while others can only generate reports.

- You can create as many users, user roles and user groups as you need.
- You can assign specific authorities to user groups or individual users via user roles.
- You can create multiple user roles each with their own authorities.
- You can assign user roles to users to grant the users the corresponding authorities.
- You can assign each user to organisation units. Then the user can enter data for the assigned organisation units.

User management terms and definitions

Term	Definition	Example
Authority	A permission to perform one or several specific tasks	Create a new data element Update an organisation unit View a report
User	A person's DHIS2 user account	admin traore guest
User role	A group of authorities	Data entry clerk System administrator Antenatal care program access
User group	A group of users	Kenya staff Feedback message recipients HIV program coordinators

You manage users, user roles and user groups in the **Users** app.

Objects in the Users app

Object type	Available functions
User	Create, edit, invite, clone, disable, display by organisation unit, delete and show details
User role	Create, edit, share, delete and show details
User group	Create, edit, join, leave, share, delete and show details

About users

Each user in DHIS2 must have a user account which is identified by a user name. You should register a first and last name for each user as well as contact information, for example an email address and a phone number.

It is important that you register the correct contact information. DHIS2 uses this information to contact users directly, for example sending emails to notify users about important events. You can also use the contact information to share for example dashboards and pivot tables.

A user in DHIS2 is associated with an organisation unit. You should assign the organisation unit where the user works.

When you create a user account for a district record officer, you should assign the district where he/she works as the organisation unit.

The assigned organisation unit affects how the user can use DHIS2:

- In the **Data Entry** app, a user can only enter data for the organisation unit she is associated with and the organisation units below that in the hierarchy. For instance, a district records officer will be able to register data for her district and the facilities below that district only.
- In the **Users** app, a user can only create new users for the organisation unit she is associated with in addition to the organisation units below that in the hierarchy.
- In the **Reports** app, a user can only view reports for her organisation unit and those below. (This is something we consider to open up to allow for comparison reports.)

An important part of user management is to control which users are allowed to create new users with which authorities. In DHIS2, you can control which users are allowed to perform this task. The key principle is that a user can only grant authorities and access to data sets that the user itself has access to. The number of users at national, province and district level are often relatively few and can be created and managed by the system administrators. If a large proportion of the facilities are entering data directly into the system, the number of users might become unwieldy. It is recommended to delegate and decentralize this task to the district officers, it will make the process more efficient and support the facility users better.

About user roles

A user role in DHIS2 is a group of authorities. An authority means the permission to perform one or more specific tasks.

A user role can contain authorities to create a new data element, update an organisation unit or view a report.

A user can have multiple user roles. If so, the user's authorities will be the sum of all authorities and data sets in the user roles. This means that you can mix and match user roles for special purposes instead of only creating new ones.

A user role is associated with a collection of data sets. This affects the **Data Entry** app: a user can only enter data for the data sets registered for his/her user role. This can be useful when, for example, you want to allow officers from health programs to enter data only for their relevant data entry forms.

Recommendations:

- Create one user role for each position within the organisation.
- Create the user roles in parallel with defining which user is doing which tasks in the system.

-
- Only give the user roles the exact authorities they need to perform their job, not more. Only those who are supposed to perform a task should have the authorities to perform it.

About user groups

A user group is a group of users. You use user groups when you set up sharing of objects or notifications, for example push reports or program notifications.

See also:

[Sharing](#)

[Manage program notifications](#)

[Manage push reports](#)

Workflow

1. Define the positions you need for your project and identify which tasks the different positions will perform.
2. Create roughly one user role for each position.
3. Create users.
4. Assign user roles to the users.
5. Assign the users to organisation units.
6. (Optional) Group users in user groups.
7. Share datasets with users or user-groups via the Sharing Dialog in Data set management section of the Maintenance app

Tip

For users to be able to enter data, you must add them to an organisational unit level and share a dataset with them.

Example: user management in a health system

In a health system, users are logically grouped with respect to the task they perform and the position they occupy.

1. Define which users should have the role as system administrators. They are often part of the national HIS division and should have full authority in the system.
2. Create roughly one user role for each position.

Examples of common positions are:

Position	Typical tasks	Recommended authorities	Comment
System administrators	Set up the basic structure (metadata) of the system.	Add, update and delete the core elements of the system, for example data elements, indicators and data sets.	<p>Only system administrators should modify metadata. If you allow users outside the system administrators team to modify the metadata, it might lead to problems with coordination.</p> <p>Updates to the system should only be performed by the administrators of the system.</p>
National health managers Province health managers	Monitor and analyse data	Access to the reports module, the Maps , Data Quality apps and the dashboard.	Don't need access to enter data, modify data elements or data sets.
National health information system division officers (HISO)	Enter data that comes from facilities which are not able to do so directly	Access to all the analysis and validation apps	-
District health records and information officers (DHRIO)	Monitor, evaluate and analyse data	Access to the Data Entry app.	
Facility health records and information officers (HRIO)			
Data entry clerks	-	-	-

Guidelines for offline data entry using DHIS 2

The de facto standard way of DHIS 2 deployment has become *online* meaning that a single instance of the application is set up on a server connected to the Internet and all users connect to the application using a web browser over internet. This was made possible thanks to the steady increase of internet availability (mostly mobile internet), the offerings of readily available and cheap cloud-computing resources combined with the fact that DHIS 2 does not request a significant bandwidth. These developments make it possible to access on-line servers in even the most rural areas using mobile Internet modems (also referred to as dongles).

This on-line deployment style has huge positive implications for the implementation process and application maintenance compared to the traditional off-line standalone style:

Hardware: Hardware requirements on the end-user side are limited to a reasonably modern computer/laptop and Internet connectivity through a fixed line or a mobile modem. There is no need for a specialized server for each user, any Internet enabled computer will be sufficient. A server will be required for on-line deployments, but since there is only one (or several) servers which need to be procured and maintained, this is significantly simpler (and cheaper) than maintaining many separate servers in disparate locations. Given that cloud-computing resources continue to steadily decrease in price while increasing in computational power, setting up a powerful server in the cloud is far cheaper than procuring hardware.

Software platform: The end users only need a web browser to connect to the on-line server. All popular operating systems today are shipped with a web browser and there is no special requirement on what type or version. This means that if severe problems such as virus infections or software corruption occur one can always resort to re-formatting and installing the computer operating system or obtain a new computer/laptop. The user can continue with data entry where it was left and no data will be lost.

Software application: The central server deployment style means that the application can be upgraded and maintained in a centralized fashion. When new versions of the applications are released with new features and bug-fixes it can be deployed to the single on-line server. All changes will then be reflected on the client side the next time end users connect over the Internet. This obviously has a huge positive impact for the process of improving the system as new features can be distributed to users immediately, all users will be accessing the same application version, and bugs and issues can be sorted out and deployed on-the-fly.

Database maintenance: Similar to the previous point, changes to the meta-data can be done on the on-line server in a centralized fashion and will automatically propagate to all clients next time they connect to the server. This effectively removes the vast issues related to maintaining an upgraded and standardized meta-data set related to the traditional off-line deployment style. It is extremely convenient for instance during the initial database development phase and during the annual database revision processes as end users will be accessing a consistent and standardized database even when changes occur frequently.

Although a DHIS 2 implementation can be labeled as online, it is worth noting that such deployment may not purely online and may have local variation depending on local constraints. For example, while most users in countries enjoy easy access to their national DHIS 2 instance using their mobile internet or better connectivity means, some unfortunately still struggle to access the system either for data entry or analysis in places where Internet connectivity is volatile or missing in long periods of time. And for these struggling users, alternative ways to interact with the system need to be found.

This guideline aims at providing advice on how to mitigate the effect of lack reliable internet in challenging settings.

Cases and corresponding solutions

In this section, we will examine possible challenging cases and describe possible ways to address them or to minimize their effects on users and the entire system on a short term. Obviously, the possible solutions proposed in this guidelines should be adapted in each context by taking into account many other parameters such as security, local practices and rules etc. The thinking in this guideline is not to prescribe bullet proof solutions that can work everywhere but propose ways of addressing connectivity issues in some places in the country.

We identify three (3) main scenarios:

1. Limited internet availability and data entry forms are small
2. Limited internet availability and data entry forms are huge
3. Internet is not at all available

We recognize that these scenarios are very simplistic because in practice a health facility can have for instance one small weekly form for disease surveillance, one big form for monthly progress report and a middle sized form for a health program. This makes the number of possible scenarios for a given setting greater than what is spelled out here. It will be therefore for up to each implementation team to discuss with the stakeholders to make simple choices that address all of the scenarios in a given setting. In most cases about 80 to 95% of districts (or health facilities if data entry is done at this level) will have the same configuration regarding internet availability and only the the remain 5 to 20% will need alternative ways to get their data in DHIS 2.

1. Limited internet availability (instability of signal or limited mobile data) and data entry forms are small

By limited internet availability, we mean case where:

- network signal is available and good but there is not enough resources to buy adequate mobile data to work continuously online
- network is good but fluctuates or is only available at a given period in the day
- network signal is weak but improves from time to time to allow connection to DHIS 2

And by data entry form small we mean data entry form having less than one hundred fields.

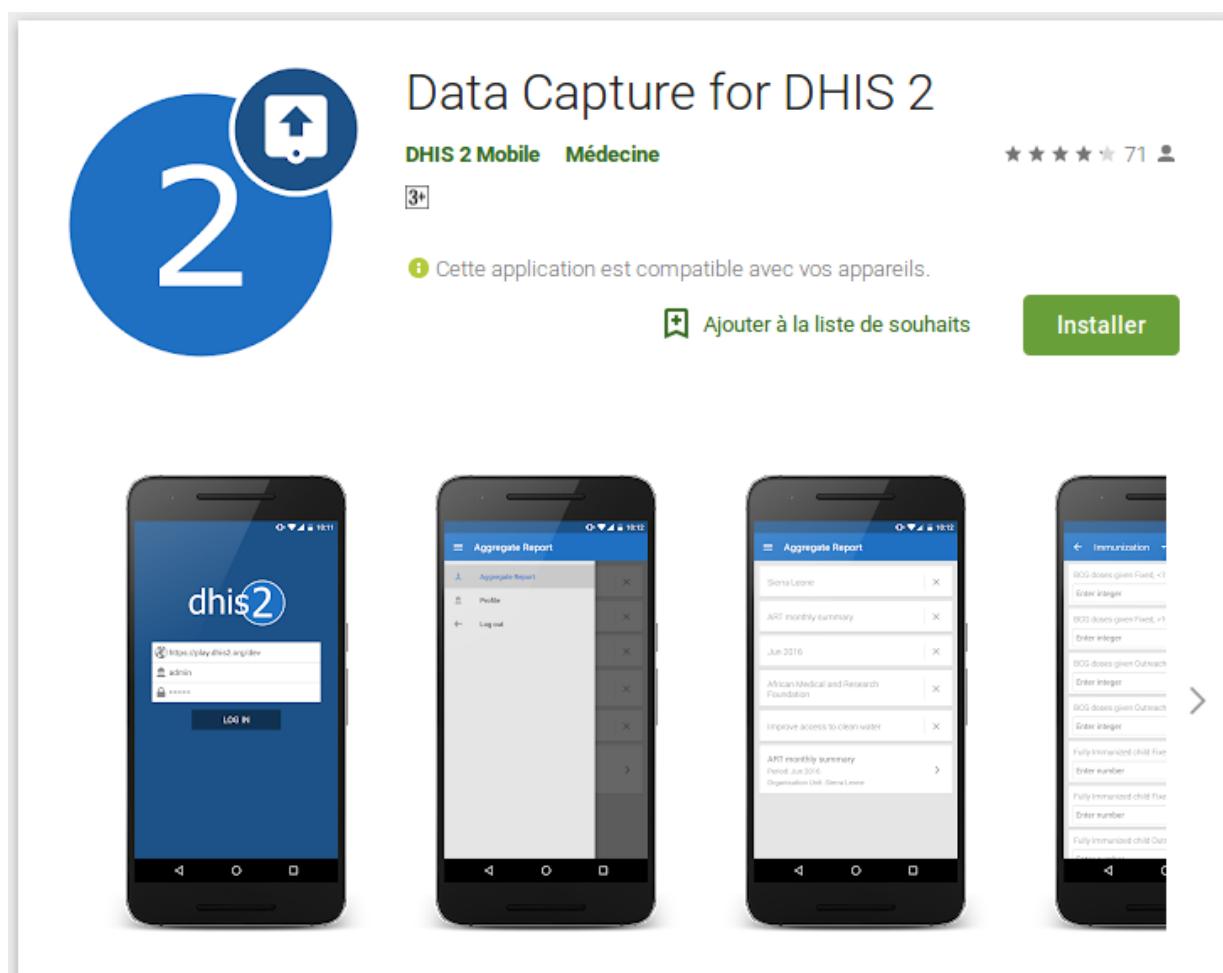
So if internet connectivity is limited and data entry forms are small, there are two possibilities to address the connectivity problem: Android data capture app and web data entry offline capability.

Use of Android data capture app:

The Data Capture for DHIS 2 app allows users to enter data into a DHIS 2 server with an Android device. The app downloads instances of forms which are required to enter data from the server, and stores them on the device. This means that users can enter data offline for facilities they are assigned to and then upload it to the DHIS 2 server when they have network coverage.

To do this, the users will be request to go to the Google Play from their Android device and type DHIS 2 data capture and get the following screen.

Then install the app called **Data Capture for DHIS 2**.



Once the app is installed and launched, the user will be requested to provide the url of their national DHIS 2, the username and password and tap LOG IN.

After a successful log in, the app will download automatically the forms and organization units the user is assigned to and store them locally for data entry. From here, any subsequent use of the app for data entry will not require internet connection as instances of forms are already stored locally. Internet connection will be needed only to sync data with the server. This can be done when internet is available locally.

On the system administration side, organizing the data entry form into sections in DHIS 2 will make data entry experience more fluid and enjoyable.

As for the synchronization, when internet connectivity is not available when needed, the user take the mobile device to the district – during the district meeting – or to the nearest area where internet is available.

Use of the offline capability of DHIS 2 web data entry module

The web data entry module is the module inside DHIS 2 allowing for data entry using the we browser. This is in DHIS 2 the regular way of data entry online. However it does have also an “offline” capability that supports the continuation of data entry even when the internet is interrupted. This means that if the user wants to do data entry at the end of the month for instant, he has to first connect to internet, log in

to DHIS 2 and open the data entry forms for at least one of the facilities he is assigned to. From this step, he can disconnect his internet and continue data entry for all his facilities and for the periods he wants as long as the data entry web page window is not closed in the web browser. After finishing the data entry, he can close the browser and even shutdown his computer. The data entered will be stored locally in the cache of the browser and the next time the user will get online and log in DHIS 2 he will be asked to click on a button to upload it.

For this case, it is possible to use either android data entry app or the semi-offline web based feature in DHIS 2 or both depending on the size of data entry forms. However, clearing the cache of the browser will result in the loss of the data stored locally. Therefore, it is recommended to not clear the cache without making sure that data locally stored is synced.

When the user is logged in and internet is cut (deliberately or not)

Filter in section	Fixed		Outreach	
	<1y	>1y	<1y	>1y
BCG doses given	23	34	45	45
Fully Immunized child	34	34	45	45
LLITN given at Penta3	23	3	23	23
OPV0 doses given	8	23	32	23
OPV1 doses given	5	23	23	23
OPV2 doses given	7	23	3	23
OPV3 doses given	6	23	3	23
PCV1 doses given	6	1	1	12
PCV2 doses given	23	23	1	2
PCV3 doses given	23	23	23	23
Penta1 doses given	5	23	23	123
Penta2 doses given	7		3	23
Penta3 doses given	6		3	23
Yellow Fever doses given	7	23		

When internet is back and the user log in DHIS 2

2. Limited internet availability and data entry forms are huge

When internet but the availability is limited but the data entry form contains several hundreds of fields, it limits possible solutions. In this case it is not advisable to use the android capture for two reasons:

- it can regularly crash because it is not designed to handle forms of very big size
- it can turn out to be tedious and eye exhausting for users because the screen is small and does not allow for fast data entry

Thus the only option available is to use the web data entry module offline capability described above or move to the nearest place where internet is available when the user cannot afford to wait the next time internet will be available in his area.

3. Internet is not at all available

In this case there are three options:

- Use of the Android capture app for data entry locally and sync the data at the upper level where internet is available if the user attends regular meetings there. This is only feasible if the forms are small
- Move to the nearest place (if affordable) or use the opportunity of regular meeting at the upper level to capture data with the web data entry module. In this case depending on the internet connectivity the user can either work online or use the offline capability described in the section [above](#).
- Ask the upper level where internet is available to do data entry regardless of the size of the form. Although this data entry happens at upper level, data can still be entered for every health facility.

Integrating tracker and aggregate data

This guide presents different approaches for combining data collected through tracker programmes with aggregate data, so that it can be analysed and used together. Tracker and aggregate data are collected and stored separately in DHIS2, but there are many cases where combining the two types of data is useful:

- Data collected through tracker programmes and aggregate data sets may be complimentary. For example, if tracker is used as an electronic immunisation registry, calculating immunisation coverages requires the service data collected through tracker to be combined with population estimates typically available as aggregate (yearly) data.
- In many cases, tracker implementations are done in a phased approach, where it is first implemented in certain types of health facilities or by geographical area. Consequently, the same data may be collected through tracker in some locations and as aggregate data in other locations, and getting a complete overview of the data requires the tracker and aggregate data to be combined. Such differentiated or hybrid approaches may also be permanent.
- Data collected through tracker may partially overlap with established aggregate reports. For example, a monthly report on malaria-related activities may include information both on malaria cases, as well as preventive activities such as bed-net distribution. If tracker is introduced for malaria case registration, the monthly malaria report can be partially completed based on tracker data, but still also require aggregate reports for preventive activites.
- When tracker is introduced in a programmatic area (immunisation, HIV etc) where aggregate data has previously been collected, ensuring that data is comparable over time necessitates combining aggregate and tracker data to allow longitudinal data analysis.
- Certain data quality checks in DHIS2 are only available for aggregate data. Applying these checks on tracker data thus requires that it is first aggregated and stored as aggregate data elements.

There are several ways in which this can be achieved, suitable for different purposes. Each of these approaches have advantages and disadvantages. In the [next section](#) three overall approaches to combining tracker and aggregate data are presented, followed by a section on [choosing an approach](#) that outlines considerations and examples of when each of the approaches may be appropriate. Then, a [how-to guide](#How-to-saving-aggregated-tracker-data-as-aggregate-data-values] is provided for the approach based on saving data from tracker as aggregate data values.

Alternative approaches

Tracker and aggregate data can be combined in DHIS2 by:

- showing tracker and aggregate data side by side in the same chart, table, map or dashboard;
- combining aggregate and tracker data through aggregate indicators;
- saving values calculated from tracker data as aggregate data values.

This section gives a summary of the three approaches, with the advantages and disadvantages of each.

Showing tracker and aggregate data side by side

Aggregate and tracker data can be shown and analysed together by including it within the same Data Visualizer charts or tables. Furthermore, visualisations of tracker-based data can be created in the Event Report and Event Visualizer apps, and combined with visualisations of aggregate data on Dashboards. Any user can access to both types of data in the DHIS2 analytics apps can use this approach.

Advantages:

- easy to set up
- works well for presenting and analysing complimentary data
- detailed data can be included (e.g. line lists of cases)

Disadvantages:

- limited dimensionality in analysis of tracker data, i.e. for showing age/sex disaggregations
- not fully integrated/comparable with aggregated data, for example the same disaggregations cannot be applied in a single visualisation even if available in the underlying data
- requires the tracker and aggregate data to be in the same DHIS2 instance

Combining data through aggregate indicators

Aggregate indicators can be based on both aggregate and tracker data, separately or combined in a single aggregate indicator. Tracker data elements, tracked entity attributes and program indicators can all be included in the calculation of aggregate indicators.

This approach can be useful in several scenarios:

- The same data is collected through aggregate data set and tracker programmes in different health facilities, i.e. some collect aggregate data and others collect individual-level data through tracker.
- The same data is available as aggregate data values and tracker data values for different periods, for example if data currently collected through tracker was in previous years collected as aggregate data.
- When indicators are needed based on a combination of data, i.e. service data collected through tracker combined with denominators available as aggregate data.

Advantages:

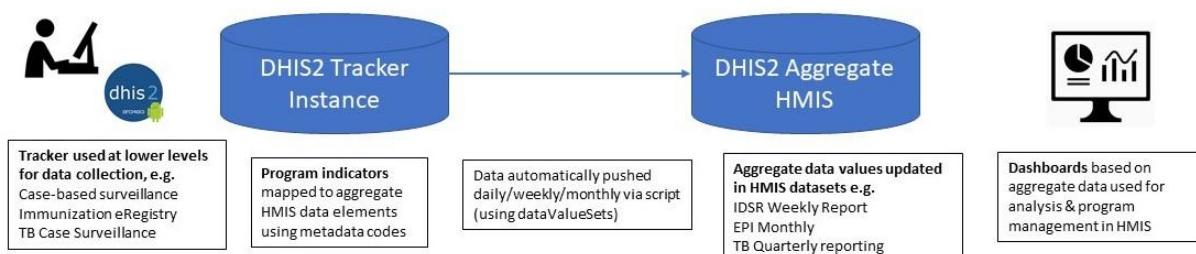
- relatively easy to set up
- can potentially hide some of the complexity of integrating aggregate and tracker data to end users

Disadvantages:

- tracker data cannot be analysed with disaggregations such as age/sex as separate data dimensions
- difficult to manage in cases where there may be overlapping data
- requires the tracker and aggregate data to be in the same DHIS2 instance

Saving aggregates of tracker data as aggregate data

Tracker data can be aggregated to, for example, weekly or monthly values, and these values can be saved as aggregate data element values in DHIS2. This corresponds to what is often done manually in health facilities when registers are tallied every month to produce monthly reports. Program indicators can be defined that produce aggregate numbers based on tracker data, corresponding to aggregate data elements. The program indicator value should represent the same value as the aggregate (e.g. *number of new and relapsed TB cases notified* or *number of BCG doses given to children under 1*). The data transfer can be done on an ad-hoc basis as needed, or as part of a routine process where data is (automatically) transferred at fixed intervals (shown in the figure below).



Example: Information flow between a DHIS2 instance with tracker programmes, and a DHIS2 HMIS instance with aggregate data.

The screenshot shows a DHIS2 interface titled "WHO Aggregate Configuration Development Instance". On the left is a navigation tree for "Trainingland" regions. The main area displays several data tables and charts:

- Organisation Unit:** Cardinal Hospital Gateway PHC, Data Set: TB case notification, Period: January - March 2020.
- All tuberculosis cases registered:**

Previous anti-TB treatment status			
New	Relapse	Previously treated (excluding relapse)	Previous treatment history unknown
5	1	1	16
5	3	1	47
5	3	7	163
- All new and relapse cases (bacteriologically confirmed or clinically diagnosed) by age group and sex:**

Age (years)		0-4	5-14	15-24	25-34	35-44	45-54	55-64	65+	Unknown
Male		11	4	0	7	35	39	31	25	13
Female		43	9	1	0	0	31	34	3	11
- Laboratory diagnostic activity:**

Patients with presumptive TB undergoing bacteriological examination	2
---	---

Example: Aggregate data set (in the data entry app) which has been automatically filled by the data pushed from tracker program indicators.

There are multiple ways in which the actual transfer of data from program indicators to aggregate data elements can be done. This includes:

- manually or via a script querying the [DHIS2 API](#) to export the program indicator values, and subsequently importing them into DHIS2 using either the [Import/export app](#) or the API;
- automating the export and import of data from the API using a script;
- using one of several applications developed by the DHIS2 community and available on the [DHIS2 App Hub](#) to export and import the data;
- setting up [Predictors](#), which can be scheduled to transfer the program indicator values into aggregate data elements routinely.

This approach is described in further detail below, focusing on how to automate the process using scripts.

Advantages:

- data can be analysed with all the same dimensionality as aggregate data (see example screenshot below)
- data can still be combined with detailed tracker data (i.e. the first approach)
- ensures that there is no overlap in tracker and aggregate data
- works when tracker and aggregate data are collected in separate DHIS2 instances
- may improve efficiencies and reduce load on the DHIS2 server from analytics, since requests for pre-aggregated data is often less demanding than on-the-fly aggregation of tracker data

Disadvantages:

- more complex to set up, and may require more ongoing maintenance
- generally requires external tools/scripts to move data via the API
- a mapping of data between tracker and aggregate data must be developed and maintained
- if data is moved between two DHIS2 instances, organisation units must also be harmonised and kept in sync across the instances

DHIS 2 Pivot Tables

Data	
Indicators	Select indicator group
Available	Selected
2.09.11 - TB case - New and relapse TB cases - 25-34 years male 2.09.12 - TB case - New and relapse TB cases - 25-34 years female 2.09.13 - TB case - New and relapse TB cases - 35-44 years male 2.09.14 - TB case - New and relapse TB cases - 35-44 years female 2.09.15 - TB case - New and relapse TB cases - 45-54 years male 2.09.16 - TB case - New and relapse TB cases - 45-54 years female 2.09.17 - TB case - New and relapse TB cases - 55-64 years male 2.09.18 - TB case - New and relapse TB cases - 55-64 years female 2.09.19 - TB case - New and relapse TB cases - 65+ years male 2.09.20 - TB case - New and relapse TB cases - 65+ years female	

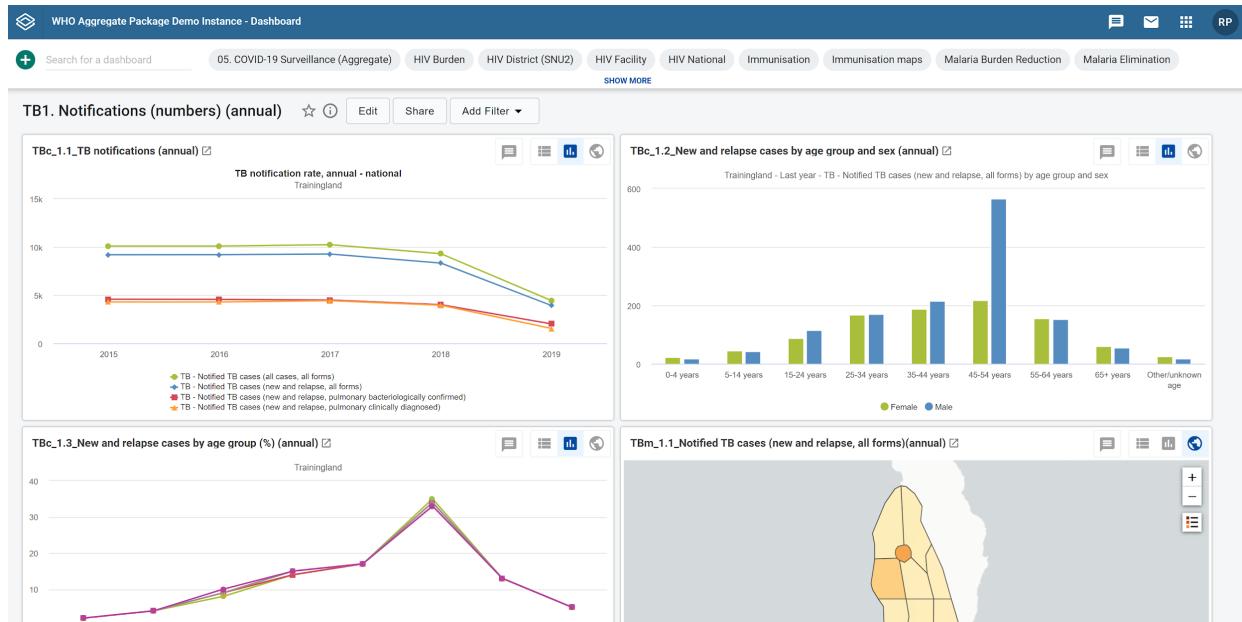
Periods	
Organisation units	Ownership
Type	Urban/Rural

*** GTB Report**

Trainingland

	2019
2.09.01 - TB case - New and relapse TB cases - 0-4 years male	0
2.09.02 - TB case - New and relapse TB cases - 0-4 years female	1
2.09.03 - TB case - New and relapse TB cases - 5-9 years male	2
2.09.04 - TB case - New and relapse TB cases - 5-9 years female	1
2.09.05 - TB case - New and relapse TB cases - 10-14 years male	1
2.09.06 - TB case - New and relapse TB cases - 10-14 years female	2
2.09.07 - TB case - New and relapse TB cases - 15-19 years male	3
2.09.08 - TB case - New and relapse TB cases - 15-19 years female	0
2.09.09 - TB case - New and relapse TB cases - 20-24 years male	1
2.09.10 - TB case - New and relapse TB cases - 20-24 years female	2
2.09.11 - TB case - New and relapse TB cases - 25-34 years male	6
2.09.12 - TB case - New and relapse TB cases - 25-34 years female	5
2.09.13 - TB case - New and relapse TB cases - 35-44 years male	1
2.09.14 - TB case - New and relapse TB cases - 35-44 years female	0
2.09.15 - TB case - New and relapse TB cases - 45-54 years male	3
2.09.16 - TB case - New and relapse TB cases - 45-54 years female	1
2.09.17 - TB case - New and relapse TB cases - 55-64 years male	0
2.09.18 - TB case - New and relapse TB cases - 55-64 years female	0
2.09.19 - TB case - New and relapse TB cases - 65+ years male	0
2.09.20 - TB case - New and relapse TB cases - 65+ years female	1

Example: TB Case Surveillance to aggregate quarterly TB report for TB notifications. Program indicator data values from tracker data (data elements/disaggregation can be produced, but not pivoted by dimensions as gender, age group)



Example: Aggregate dashboard output (with ability to pivot male/female as a data dimension)

Choosing an approach

Each of the three approaches for has advantages and disadvantages, as outlined above. For a single implementation, several of them may be needed. For example, it may be useful to present certain

tracker data with frequent updates (i.e. daily numbers children immunised), while at the same time transferring aggregate program indicator values into aggregate data element values every month so that the data can be compared with facilities not yet using tracker, or with additional dimensionality (such as age/sex disaggregations) that cannot easily be done directly with aggregate data.

The first two approaches are both relatively straightforward to implement, using the standard applications built into DHIS2. While configuring aggregate indicators (the second approach) must be done by a system administrator with access to configure such indicators, any user with access to the DHIS2 analysis apps and the data itself can use these approaches. However, a major limitation is that they require the tracker and the aggregate data to be in the same instance of DHIS2.

The third approach, saving data from tracker as aggregate data values, has some advantages in terms of analytics. However, it is also the only approach suitable for integrating tracker data with aggregate data in separate DHIS2 instances. Many countries have a mature, stable DHIS2 instance used primarily for capturing aggregate data across health programs in an integrated environment (e.g. a Health Management Information Systems, HMIS). When implementing DHIS2 tracker for individual-level data collection, doing this in a separate DHIS2 instance dedicated to the tracker deployment is generally recommended. By maintaining separate tracker and aggregate DHIS2 instances, performance can be better managed by system admins, DHIS2 updates can be performed independently, and data governance principles can be applied to ensure personally identifiable data captured by tracker can be protected according to national policies and governance frameworks.

When a system for routine aggregate reporting through DHIS2 data exists, there is a clear benefit in being able to leverage individual data collection through tracker to automatically 'report' aggregated data to the routine HMIS. The alternative is often that this is done manually by the health facilities, since such aggregate summaries are important for the management of the individual health facilities, and routine reporting of aggregate data is often mandatory. Capturing individual level data through DHIS2 tracker has potential to improve the quality of the data reported into the routine aggregate HMIS, while also enabling ad hoc analysis of the individual-level tracker data as required.

There are several ways, technically, to do this. In the how-to section below, the focus is on the steps needed to set up an automated migration of tracker data into aggregate data values.

How-to: saving aggregated tracker data as aggregate data values

This section describes the recommended approach for saving tracker data as aggregate data element values. While requiring an external tool or script as part of transferring the data, it leverages the existing functionality of DHIS2 as far as possible so that the script can be relatively simple. What is outlined here is also the approach taken in the [WHO configuration packages](#) for DHIS2, which includes mapping of variables between tracker programmes and aggregate data sets where relevant. This is discussed in more detail [below](#).

The approach described here is recommended as a long-term, automated solution for saving tracker data as aggregate data values. Technically, there are several other ways in which this aggregation and migration of data can be done, including using predictors, exporting the data and transforming it using other software (for example excel), or through custom DHIS2 apps (including some that are available in the [DHIS2 App Hub](#)). While not described in this guide, these tools and methods can still be relevant in many cases, including in combination with the approach outlined here. For example, if there is only a need to do ad-hoc transfers of data from time to time, or in an early phase of tracker implementations when data is transferred primarily for testing and the configuration is still undergoing changes.

Implementation considerations

Integrating data collected through tracker with existing aggregate reporting flows (i.e. the HMIS) requires decisions to be made related to data governance, and it affects data access, systems maintenance and more. Some key considerations are presented here.

Data transfer

There are two key considerations related to the transfer of data:

How often should aggregated data be transferred from tracker to aggregate data values? When the transfer is automated, the frequency of the transfer can be anything from daily to only once per reporting/aggregation period (e.g. weekly, monthly, quarterly). More frequent updates means that data becomes available as aggregate data values and can be used and analysed more quickly, and are kept up to date as new and updated information comes in. Whether or not this is useful depends on the tracker programme in question. For example, having daily updated data over the course of a reporting period may be useful information if made available to facility-level staff, but is less useful if the purpose of the aggregation is primarily to facilitate and automate routine HMIS reporting to higher levels.

How far back in time should data be added and updated? Together with a decision on how frequently to transfer data, it must be decided for how far back (how many periods) data should be updated, and whether or not to transfer data for the current period (for which data will not be complete, see previous point). This decision may have to be aligned with potentially existing practices around aggregate data, such as when or how it is validated and whether or not it is at some point locked for editing, discussed further below. Related to this is the question of whether to differentiate between migrating new data values and making updates to previously reported values.

Discussions on these issues need to take into account that tracker data is in many cases entered retroactively based on paper registers, rather than directly during service provision or patient encounters. Furthermore, corrections and edits may happen to the data for quite some time after the actual event took place, for example, if during a follow-up visit an error is detected in the data for the previous visit.

Unless there are strong reasons to do otherwise, it is suggested that updates and edits are done as far back in time as there is reasonable chance of additions and updates being made to the underlying tracker data. This ensures that the most correct and updated information is what is used, even though it may necessitate changes to HMIS data management standards around e.g. data validation and locking.

Data quality and validation

Ensuring data quality is a key concern both for tracker and aggregate data, and linking the two introduces new potential dilemmas in this area. There are tools and methods for reducing the chance of errors being introduced in tracker data collection. Nonetheless, there is always a chance that errors are introduced. Within the period in which the tracker-based aggregate data is still being updated on a regular basis (as described in the previous section), corrections in the tracker data flows automatically to the aggregate data as well. However, there are two scenarios for which it must be decided how to address corrections to the data:

If errors are detected and corrected in tracker, after the time period in which data is routinely migrated and/or after which the aggregate data has been validated and/or locked. Possible ways to address this includes:

- living with the discrepancy in the aggregate data (if the error is minor);
- doing an ad-hoc transfer of the data for affected periods;
- manually correcting the aggregate data.

If data quality issues are detected in the aggregate data. This is a less likely scenario, since only relatively large or systematic errors in the tracker data will be visible when the information is aggregated, or if the data is obviously incomplete. Possible ways to address this includes:

- correcting the source data in tracker and then re-transferring the data (if possible);

-
- correcting/editing the aggregate data (if possible), and accepting the discrepancy.

Another data quality topic of relevance when aggregating tracker data relates to the timeliness and completeness of data, which are key data quality metrics of aggregate reporting (e.g. in the HMIS). When aggregate data is reported directly through DHIS2, users click a button to indicate that a particular data set (reporting form) has been reported in full. This is used as the basis for calculating both the timeliness (submissions by a specified deadline) and completeness of data. When aggregate data is generated based on tracker data, no completeness and timeliness information is available.

Several approaches can be considered concerning this issue:

- In some cases, it is unproblematic that there is no completeness and timeliness data. There is generally no completeness and timeliness information for tracker data, and the aggregate data values generated from tracker can be seen in the same way. This is the case, for example, if data is transferred primarily to facilitate data analysis with additional dimensionality, or in order to use analysis tools for aggregate data.
- If the data makes up a subset of a particular routinely reported data set, where other parts are entered directly as aggregate data, completeness information for the tracker data could be verified and reported as part of the completeness of the overall dataset.
- Completeness and timeliness information can be managed manually, by the user responsible for submitting the aggregate data. This can be done as part of a validation process, where the user verifies the data (in the aggregate data entry app) and then confirms that the data is complete. While this allows for an extra validation step, it is also more resource intensive, and true validation of the data would to some extent require a degree of manual tallying that partly defeats the purpose of automating the aggregation of tracker data.
- A script or tool can relatively easily be developed to automatically mark data sets as complete if a certain amount (i.e. a specified number of data element values) has been reported. This works well for identifying health facilities for which *some* data has been reported, but this automated process cannot determine whether the reports are in fact *complete* in the true sense of the word.

Generally, when tracker data is used to produce aggregate data values, this is a form of secondary use of data beyond what it was primarily collected in order to do. It is important that it is clearly communicated to the users how issues of data validation and corrections are managed, how issues such as "completeness" are dealt with, and that the "source of truth" for the data is clearly defined.

Data access and ownership

Access to both tracker and aggregate data in DHIS2 is controlled through sharing, based on user groups. Sharing of tracker data and of the aggregate data values generated from the tracker data can thus be different, and in the common scenario that these two types of data are hosted in different DHIS2 instances the same users may not have access to the system at all. This has certain advantages, for example, the aggregate data values may be shared more widely than the tracker data without privacy/security implications. At the same time, it requires that appropriate data sharing is set up in two different instances, and it may also necessitate that users access two different systems. (It is possible to use [OpenID Connect](#) to allow users to share username and password across the two instances.)

Related to data access is the issue of ownership of data, an issue also linked to data quality and validation. There needs to be clear procedures in place that indicate who are responsible and "own" both the tracker and the aggregate data values generated from tracker. This is particularly important in scenarios where multiple health programmes are involved. For example, if an immunisation tracker programme feeds data into an integrated, aggregate HMIS data set for which a separate HMIS unit is responsible.

Maintenance

The approach described here requires that technical capacity is available, both to initially develop and configure a solution for data transfer, and for ongoing maintenance that may be required for example if there are changes to the DHIS2 server infrastructure. Furthermore, if metadata in the tracker instance or in the aggregate HMIS instance is altered, the mapping of program indicators to aggregate data elements may need to be updated accordingly.

Transition period

When tracker data is aggregated for the purpose of replacing established aggregate reporting (such as existing routine HMIS reports), it is often useful to plan for maintaining parallel reporting for a period of time, for example 6 months. In this period, aggregate numbers generated from tracker and from the existing manual reporting procedures should be compared. It is unlikely that they will ever be completely identical, but such comparisons are useful because they:

- Should trigger a discussion around the source of the discrepancies, for example where there are data quality problems (in either of the data sources).
- Informs decisions around when the tracker data is of the same or better completeness and quality as the manual reporting, so that the parallel reporting can be stopped.

Technically, this can be achieved by having a separate "shadow" data set with separate data elements in the aggregate instance, so that two parallel sets of aggregate data can be kept and compared there. Alternatively, a copy of the aggregate data set can be kept in the tracker instance and used for comparisons.

Assumptions and key steps

When tracker data and aggregate data are managed in separate DHIS2 instances, which is generally recommended, migrating data between the two instances requires the organisation units to be the same, or at least have a shared set of identifiers. Because organisation units are often re-used when new DHIS2 instances are set up, this may not be a problem initially. However, keeping organisation units synchronised across two or more DHIS2 instances over time requires careful management, whether changes are handled manually or through an automated process. Forthcoming implementation guidance will discuss this issue in further detail. For the purpose of this guide, a prerequisite is that organisation units are harmonised and have a shared set of identifiers across the two instances. In cases where tracker data is migrated to aggregate data values within the same DHIS2 instances, synchronising organisation units is not an issue.

Conceptually, the steps involved in migrating aggregates of tracker data to aggregate data values involves:

1. Establish a mapping between program indicators and the corresponding data elements and category option combinations, using codes.
2. Export program indicator values as an aggregate *data value set*
3. Import the *data value set* as aggregate data element values

The following sections explains (1) how to establish a mapping between program indicators and data elements, (2) the DHIS2 APIs relevant for importing/exporting data values, and (3) considerations when automating the import and export process.

Mapping program indicators with aggregate data elements

To produce aggregate data values from tracker data, program indicators must be defined for each data point. Each of these data values corresponds to a data element and if applicable a category option combination. A mapping using an identifier is thus needed to specify which program indicator relates to which specific data element (with category option combination). While not discussed in detail here, the mapping can also include attribute option combinations.

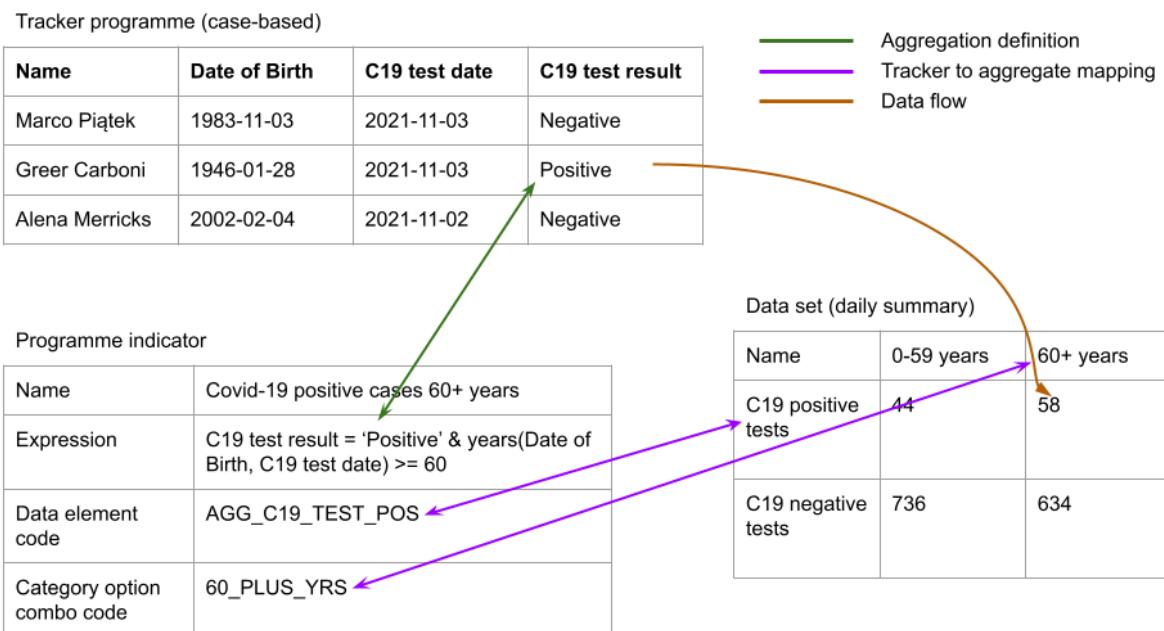


Illustration of how tracker data is mapped to aggregate data using program indicators.

It is recommended to use codes as identifier for this mapping, and this is what is presented here.

Note While the approach described here is based on the mapping of data being done in the source system (through program indicators), it could in other scenarios be done in the target system where the data is imported, or in a middleware or interoperability layer in between.

Coding data elements

Because the data element and category option combination codes will be added as attributes to the program indicators, the first step is to create and/or add a code to the data elements and category option combinations. This should be done in the DHIS2 instance in which aggregate data values will be saved. If the data elements and category option combinations already have codes, these can be used. It is also possible to define custom attributes and assigned these to the data elements for the purpose of mapping, but for the purpose of this guide we will use the built-in data element code.

While not strictly necessary, it may be advisable to add the data elements to a data set if they are not already. This data set (or sets) define the period type of the data to be transferred (e.g. weekly, monthly), and when imported the aggregated tracker data will be validated against this period type. Furthermore, such assigned may be the basis of subsequent calculation of data completeness.

Coding program indicators

Program indicators have a fixed attribute used specifically for the category option combination (and attribute option combination) identifier that is used when the program indicator value is exported as a data values set.

Category option combination for aggregate data export

Attribute option combination for aggregate data export

SAVE

CANCEL

Program indicator fields for category option combination and attribute option combination

However, there is by default no corresponding field for which to specify the identifier (i.e. code) of the data element. In principle, the code of the program indicator itself *could* be used, but this will fail in the common scenario that multiple program indicators are linked to the same data element (with different category option combinations). Instead, it is recommended to create an [attribute](#) that is assigned to program indicators.

The custom attribute should be of the type Text. It should *not* be mandatory, since not all program indicators will be linked to aggregate data elements. It should *not* be unique, since multiple program indicators may point to the same data element code. And it should be applied to "Program indicator" only, since it is not relevant elsewhere. Other properties like name, description and code can be defined according to the metadata naming convention of the particular implementation. The screenshot below shows how the custom attribute included in the WHO metadata packages is configured. If this custom attribute has already been imported into the DHIS2 instance in question, it can be re-used.

← Attribute ⓘ

The screenshot shows the 'Attribute' creation form in the DHIS2 interface. The 'Name (*)' field contains 'Data element for aggregate data export'. The 'Short name' field also contains 'Data element for aggregate data export'. The 'Code' field is set to 'DATA_ELEMENT_AGG'. The 'Description' field is also 'Data element for aggregate data export'. The 'Value type (*)' field is circled in orange and has 'Text' selected. The 'Option set' section includes icons for edit, add, and delete. The 'Sort order' section includes checkboxes for 'Mandatory' and 'Unique', both of which are circled in orange. Below these, a note says 'The custom attribute will be applied to selected objects.' followed by three checkboxes: 'Category', 'Document', and 'Program indicator', with 'Program indicator' checked and circled in orange.

Name (*)
Data element for aggregate data export

Short name
Data element for aggregate data export

Code
DATA_ELEMENT_AGG

Description
Data element for aggregate data export

Value type (*)
Text

Option set

Mandatory

Unique

The custom attribute will be applied to selected objects.

Category Document Program indicator

Example showing how a custom attribute for linking program indicators and aggregate data elements can be defined.

Once the custom attribute is assigned to program indicators, it will appear as a new field/attribute when adding or editing program indicators in the Maintenance app. Every program indicator for which data is to be transferred to aggregate data elements needs to be created and/or modified to include the code of the corresponding data element and category option combination code.

Adding the program indicators to be migrated to a program indicator group can be useful to manage large numbers of program indicators. For example, a script for migrating data can target all program indicators in a program indicator group, simplifying the configuration.

DHIS2 Web API for export and import of dataValueSets

Once the mapping between program indicators and the corresponding data elements and category option combinations has been done, aggregated program indicator data can be exported as a dataValueSet from the analytics API endpoint and subsequently imported as aggregate data values. As noted in the introduction, we assume for the purpose of this guide that organisation units are identical or have a common identifier in the DHIS2 instance where data is to be imported and exported.

Export

To export data from the DHIS2 Web API, the `/api/analytics/dataValueSet` endpoint is used, described in further detail in the [developer documentation](#). The endpoint can return data in JSON or XML representation. It requires three parameters to be specified:

- Data (dx) - which are the program indicators for which data is to be exported.
- Period (pe) - the period for which to export data for, which should match the period type of the aggregate data elements to which data should be migrated.
- Organisation unit (ou) - the organisation units for which to export data.

The format of the these parameters are described in detail in the [developer documentation](#).

In addition to specifying the data, period and organisation units dimensions of the query, it is also necessary to specify that the custom attributes holding the data element codes should be used as identifier for the data values in the data value set being exported. This is done by setting the optional `outputIdScheme` parameter to point to the UID of the custom attribute. Objects without this attribute (e.g. organisation units) will fall back to using UIDs.

A complete API call can thus look like this:

```
/api/analytics/dataValueSet.json?dimension=dx:Uvn6LCg7dVU;0diHJayrsKo&dimension=pe:LAST_MONTH  
&dimension=ou:lc3eMKXaEfW&outputIdScheme=ATTRIBUTE:vudyDP7jUy5
```

This query will return a file (in JSON format in this example) with aggregate data values.

As specified in documentation on the `/analytics/dataValueSet` API endpoint, when using attribute-based identifier schemes for export (as in this guide) there is a risk of producing duplicate data values. This can happen if multiple program indicators have been assigned the same combination of data element and category option combination codes. The boolean query parameter `duplicatesOnly` can be used for debugging purposes to return only duplicates data values, and this check is recommended after any changes to the mapping between program indicators and aggregate data elements.

```
/api/analytics/dataValueSet.json?dimension=dx:Uvn6LCg7dVU;0diHJayrsKo  
&dimension=pe:LAST_MONTH&dimension=ou:lc3eMKXaEfW&duplicatesOnly=true
```

Because the export relies on the analytics API, only data included in the [analytics tables](#) are included. So for example, if the analytics tables are scheduled to update at midnight every day, and the transfer is scheduled for 23:00 every day, data from the current day will not be included.

Import

When a file with aggregated data values has been exported from the `/api/analytics/dataValueSet` endpoint with the appropriate parameters as described above, it can be imported directly into the DHIS2 instance to which data is to be migrated. For testing purposes, the data file can be imported using the [Import/Export app](#), or via the DHIS2 Web API. Here, we show how to use the API.

The API endpoint for importing data value sets is '/api/dataValueSets', described in further detail in the [developer documentation](#). Of particular relevance is the different [import parameters](#), which must be adapted for our purposes as described here:

- `dataElementIdScheme` and `categoryOptionComboIdScheme` must be set to `code`, since we use codes to map values from Program indicators to aggregate data elements and category option combinations.
- `orgUnitIdScheme` must be set to whatever is appropriate for each particular case.
- `dryRun` makes it possible to get an import summary without actually importing data, and can be useful for testing.
- `importStrategy` should in most cases be set to `CREATE_AND_UPDATE`, which means that new data values will be imported and existing ones updated. In some cases, it *may* be relevant to set this to only `CREATE` or `UPDATE`, for example if a decision is made not to update existing data after a certain time period, but allow new data to be added. This was discussed in further detail under [implementation considerations](#)

This is an example of how to specify the appropriate parameters when importing data using the API:

```
/api/dataValueSets/  
dataElementIdScheme=CODE&categoryOptionComboIdScheme=CODE&importStrategy=CREATE_AND_UPDATE&dryRun=false
```

Writing scripts to automate data migration

A template script for automating routine migration of data from tracker to aggregate, within the same instance or across separate DHIS2 instances, will be made available in the near future. Whether adapting this template, or developing custom tools or script to perform the migration, there are certain recommendations that should be followed.

- Separate the script performing the export and import from the configuration of what data to migrate. This allows more easily modifying the configuration without danger of introducing errors in the logic of the script, and makes it easy to have multiple configurations (i.e. one for each tracker programme).
- The script should produce a log with key events and information, such as when the script has been triggered, a summary of the import results (on success), or details of the error (in case of failure).
- A system should be in place to notify the persons responsible for the data migration in case of errors, for example through an email server configured on the server, or using the messaging functionality of DHIS2 itself (accessible through the Web API).
- Because the data export relies on the analytics endpoint, it may be useful for the script to trigger analytics and performing the export after this process has finished.

DHIS2 Digital Data Packages and linking tracker and aggregate data

DHIS2 digital data packages have been developed to support both aggregate reporting & analysis, as well as tracker data capture and facility-level analysis. Further information is available on [dhis2.org/who](#)

Aggregate digital data packages (inclusive of standard aggregate dashboards) are available for health programmes such as TB, HIV, malaria, RMNCAH and disease surveillance. Aggregate packages include:

1. Data set, data elements and category option sets ('target' for sending tracker data)
2. Metadata codes that are ADX compliant and enable the mapping of data values from tracker to the aggregate 'target'

In addition, tracker packages are being developed for a growing number of use cases such as immunisation eRegistries and case-based surveillance for TB, HIV and integrated disease reporting. Where tracker data packages are designed to capture data that can be aggregated and submitted to the corresponding aggregate dataset, we have included the following in the tracker digital data packages:

1. Program indicators configured to produce data values corresponding to the data elements and disaggregations included in the aggregate digital data package data.
2. Custom attribute for 'Aggregate data element code'.
3. Attributes per program indicator populated with the data element codes and category option combination code from the aggregate digital data package.

Summary

Steps to migrate tracker data to aggregate data values:

1. Define a list of the variables for which data is to be generated and transferred.
2. Ensure that the aggregate data elements to which data will be associated exist and have a code.
3. Ensure that the category option combinations assigned to these data elements have a code.
4. Create a custom attribute assigned to program indicators.
5. Ensure that the program indicators producing the aggregate data values exists, and assign them with the code of the corresponding data element and category option combination.
6. Export program indicators values as a data value set using the /api/analytics/dataValueSet Web API endpoint in the DHIS2 instance hosting the tracker data.
7. Import the data value set to the /api/dataValueSet Web API endpoint in the DHIS2 instance that will hold the aggregate data.

Known issues

- There is a bug in earlier 2.33-36 releases that prevents custom attributes from being exposed in the UI ([Jira 8755](#)). This is resolved in the latests DHIS2 patch releases
- There is a bug ([Jira 8868](#)) that causes metadata-dependency-export tool to fail when custom attributes are assigned to program indicators.

Data Analysis Tools Overview

This chapter offers an overview of the available tools for data analysis provided by DHIS2 along with a description of the purpose and benefits of each. If you are looking for a detailed guide on how to use each tool we recommend to continue to read the user guide after finishing this chapter. The following list shows the various tools:

1. Standard reports
2. Data set reports
3. Data completeness reports
4. Static reports
5. Organisation unit distribution reports
6. Report tables
7. Charts
8. Web Pivot table
9. GIS

Data analysis tools

The following section gives a description of each tool.

Standard reports

Standard reports are reports with predefined designs. This means that the reports are easily accessible with a few clicks and can be consumed by users at all levels of experience. The report can contain statistics in the form of tables and charts and can be tailored to suit most requirements. The report solution in DHIS2 is based on JasperReports and reports are most often designed with the iReport report designer. Even though the report design is fixed, data can be dynamically loaded into the report based on any organisation unit from within the hierarchy and with a variety of time periods.

Data set reports

Data set reports displays the design of data entry forms as a report populated with aggregated data (as opposed to captured low-level data). This report is easily accessible for all types of users and gives quick access to aggregate data. There is often a legacy requirement for viewing data entry forms as reports which this tool efficiently provides for. The data set report supports all types of data entry forms including section and custom forms.

Data completeness report

The data completeness report produces statistics for the degree of completeness of data entry forms. The statistical data can be analysed per individual data sets or per a list of organisation units with a common parent in the hierarchy. It provides a percentage value for the total completeness and for the completeness of timely submissions. One can use various definitions of completeness as basis for the statistics: First based on number of data sets marked manually as complete by the user entering data. Second based on whether all data element defined as compulsory are being filled in for a data set. Third based on the percentage of number of values filled over the total number of values in a data set.

Static reports

Static reports provides two methods for linking to existing resources in the user interface. First it provides the possibility to link to a resource on the Internet through a URL. Second it provides the possibility to upload files to the system and link to those files. The type of files to upload can be any kind of document, image or video. Useful examples of documents to link to are health surveys, policy documents and annual plans. URLs can point to relevant web sites such as the Ministry of Health home page, sources of health related information. In addition it can be used as an interface to third-party web based analysis tools by pointing at specific resources. One example is pointing a URL to a report served by the BIRT reporting framework.

Organisation unit distribution reports

The organisation unit distribution report provides statistics on the facilities (organisation units) in the hierarchy based on their classification. The classification is based on organisation unit groups and group sets. For instance facilities can be classified by type through assignment to the relevant group from the group set for organisation unit type. The distribution report produces the number of facilities for each class and can be generated for all organisation units and for all group sets in the system.

Report tables

Report tables are reports based on aggregated data in a tabular format. A report table can be used as a stand-alone report or can be used as data source for a more sophisticated standard report design. The tabular format can be cross-tabulated with any number of dimensions appearing as columns. It can contain indicator and data element aggregate data as well as completeness data for data sets. It can contain relative periods which enables the report to be reused over time. It can contain user selectable parameters for organisation units and periods to enable the report to be reused for all organisation units in the hierarchy. The report table can be limited to the top results and sorted ascending or descending. When generated the report table data can be downloaded as PDF, Excel workbook, CSV file and Jasper report.

Charts

The chart component offers a wide variety of charts including the standard bar, line and pie charts. The charts can contain indicators, data elements, periods and organisation units on both the x and y axis as well as a fixed horizontal target line. Charts can be viewed directly or as part of the dashboard as will be explained later.

Web Pivot tables

The web pivot table offers quick access to statistical data in a tabular format and provides the ability to "pivot" any number of the dimensions such as indicators, data elements, organisation units and periods to appear on columns and rows in order to create tailored views. Each cell in the table can be visualized as a bar chart.

GIS

The GIS module gives the ability to visualize aggregate data on maps. The GIS module can provide thematic mapping of polygons such as provinces and districts and of points such as facilities in separate layers. The mentioned layers can be displayed together and be combined with custom overlays. Such map views can be easily navigated back in history, saved for easy access at a later stage and saved to disk as an image file. The GIS module provides automatic and fixed class breaks for thematic mapping, predefined and automatic legend sets, ability to display labels (names) for the geographical elements and the ability to measure the distance between points in the map. Mapping can be viewed for any indicator or data element and for any level in the organisation unit hierarchy. There is also a special layer for displaying facilities on the map where each one is represented with a symbol based on its type.

Localization of DHIS 2

DHIS 2 localization concepts

Localization involves the adaptation of an application to a specific location. When implementing DHIS 2 in a given country, adequate resources should be allocated to translate and localize the application if required. Translation of the user interface elements, messages, layout, date and time formats, currency and other aspects must be considered. In addition to translation of the user interface itself, metadata content which is contained in the database must also be considered to be translated.

Interface translations are compiled into the system itself, such that new translations can only be accessed by taking a newer version of DHIS 2. Database translations, on the other hand, are specific to your implementation and can be added to your existing DHIS 2 instance.

These two aspects are managed independently and the processes and tools are outlined below.

User Interface localization

Overview

DHIS 2 supports internationalization (i18n) of the user interface through the use of Java property strings and PO files. Java property files are used when messages originate from the back-end Java server, while PO files are used for front-end apps written in JavaScript. The DHIS 2 Android apps use a specific XML format.

Note

The translator need not worry about the different resource file formats; the translation platform hides the details, and only displays the strings that require translation.

For example, the figure below shows the source and target strings when translating a resource to French.

<input checked="" type="checkbox"/> 1 List and delete SMS which were sent or 2 scheduled in the system.	Afficher et supprimer les SMS envoyés ou prévus dans le système <input type="checkbox"/>
<input checked="" type="checkbox"/> 3 List and delete SMS which the system 4 received from other system.	Afficher et supprimer les SMS reçus depuis un autre système <input type="checkbox"/>
<input checked="" type="checkbox"/> 5 Create, update, view and delete command of 6 SMS received.	Commandes pour créer, mettre à jour, afficher et supprimer les SMS reçus <input type="checkbox"/>
<input checked="" type="checkbox"/> 7 J2ME Client Update Configuration	Configuration de mise à jour de client J2ME <input type="checkbox"/>
<input type="radio"/> 8 Send SMS to raw recipient	<input type="checkbox"/>
<input checked="" type="checkbox"/> 9 Person Mobile Settings	Paramètres de mobile pour les personnes <input type="checkbox"/>

There should always be an English string for all messages in DHIS 2. When the user selects a given language, and a translation is present in that language, then the translation will be shown. However, if the string in the desired language is missing then fallback rules will be applied. In cases when two given translations, such as Portuguese and Brazilian Portuguese share common messages, it is not required to perform a full translation in the variant language. Only messages which are different should be translated.

Fallback rules are then applied in the following manner (assuming the user has chooses Brazilian Portuguese as their language):

1. Display the message in Brazilian Portuguese if it exists.

2. If it does not exist in the variant language, then use the Portuguese message, if it exists.
3. If there is no message in either the base language or the variant language, choose the ultimate fallback language, English.

Important

There are a number of source strings such as "dd MMM yyyy 'to' " which are used for date/time formatting in various parts of DHIS 2. Part of the value should not be translated because it is actually a special formatting field used by either Java or JavaScript to interpolate or format a string. In this example the part of the value which **can** be translated would be "to", for instance to "a" in Spanish. The special string which should **not** be translated is "dd MMM yyyy". If these date format template strings are translated, it may result in errors in the application!

Important

Some special variables (e.g. {0}) use curly brackets. This denotes a variable which will be replaced by a number or other value by the application. You must place this variable notation in the correct position and be sure not to modify it.

Translation Platform

DHIS2 is now using [transifex](#) as our main platform for managing translations. You can access the DHIS2 resources at [translate.dhis2.org](#), or directly at <https://www.transifex.com/hisp-uio/public>.

How do I contribute to translations?

Register as a translator

The first step is to get access to the project. There are two ways to do this:

1. Navigate to the platform and create an account with transifex, then - request access to our organisation "HISP UiO" as a member of the "DHIS 2 Core Apps" translation team.
Transifex have some useful instructions here: [Getting Started as a Translator](#)
2. Email the DHIS 2 team at translate@dhis2.org to request access.
Please provide: - the name, email address and translation language of the user(s) you would like us to give access to, and - a little bit of information about why you are interested in contributing to the DHIS 2 translations

Edit translations

Once you have access as a translator, you can start translating through the transifex Web Editor.

Transifex have a useful guide here: [Translating Online with the Web Editor](#)

As far as possible, the projects represent DHIS 2 apps one-to-one. For example, the **APP: Data Visualizer** project contains the translation strings for the Data Visualizer app.

Our transifex projects for DHIS2 User Interface all start with one of the following:

- **APP:** indicates that the project contains strings for a specific app
- **APP-COMPONENT:** indicates that the project is a component library used by the apps
- **ANDROID:** indicates that the project is an Andriod app

In addition, **APP: Server-Side Resources** contains some strings that are used by several apps; namely:

- "Data Entry"
- "Maintenance"
- "Pivot Tables"
- "Reports"

Tip

To ensure that there are full translations for a particular app, e.g. "Tracker Capture", you need to make sure translations are complete in:

- The app project: **APP: Tracker Capture**
- Any projects starting with **APP-COMPONENTS**
- **APP: Server-Side Resources**, if required. For Tracker Capture this is **not required**.

Within the projects we have resources, which represent localization files in the source code. In order to support multiple DHIS2 versions, with the same localization files, the *version* is associated with each instance of the file. So, for **APP: Data Visualizer** the list of resources looks like this in the Web Editor:

i.e. there is only one source resource for the app (en.pot), but we have added the versions from 2.31 (v31) up to the latest development (master). The version is shown in the "Category" field, and is also visible as a prefix to the resource name, e.g. v31--en-pot.

Note

In general, we request translators focus on the "**master**" resource; it usually contains all strings from previous versions, and when translations are added the platform will fill in matching translations on the previous versions too.

When will new translations be available in the system?

We have a nightly service that pulls new translations from the transifex platform and opens a pull request on the source code.

The service loops over all projects and supported languages and does the following:

1. Pulls localization files from transifex (**where translations are more than 20% complete**)
2. Raises a pull request on the source code if changes are found for the language

The pull requests are reviewed and merged into the code base as part of the normal development process.

Info

The translations added to transifex will, in general, be in the next available stable release for all supported DHIS 2 versions

If you need to ensure that your translations are in the next stable release, contact us (translate@dhis2.org) explaining your needs, and we'll let you know what we can do.

Tip

The translations you add in transifex should be visible in all development demo versions on our play server (<https://play.dhis2.org>) within a few days, in most cases.

How do I add a new language?

Please contact us via email translate@dhis2.org, or on the [Community of Practice](#) and we'll add that language to the projects on transifex.

Once resources for that language are more than 20% translated, they will start to be pulled into the system. They will then become visible in the development demo versions, and be available in future releases.

Note

DHIS 2 manages metadata (database) locales independently from the UI.
See the [following section](#).

Metadata/Database translations

In addition to translation of the user interface, DHIS 2 also supports the localization of the metadata content in the database. It is possible to translate individual objects through the **Maintenance app**, but in order to better support a standard translation workflow, a specialized app has been developed for this purpose.

New metadata locales can be added in **Maintenance app > Locales**.

DHIS 2 Translations app

The DHIS 2 **Translation app** can be used to translate all metadata (data elements, categories, organization units, etc) into any locale which is present in the database.

To get started, simply choose the **Translations app** from the top level menu.

The screenshot shows the DHIS 2 Translations app interface. At the top, there are filters for 'Object' (set to 'Data Element'), 'Filter by' (set to 'All'), 'Target locale' (set to 'French'), and a search bar. A red note '1. Choose object type' is overlaid on the 'Object' filter. Another red note '2. Be sure target locale is set correctly.' is overlaid on the 'Target locale' dropdown. Below the filters, a table lists a single metadata object: 'ACCUTE FLACCID PARALYSIS (DEATHS < 5 YRS)' (Description) and 'PARALYSES FLACIDES AIGUES (DÉCÈS < 5 ANS)' (Name). To the right of the table, a red note '3. Translate a specific metadata object.' is overlaid. At the bottom left is a blue 'SAVE' button, and at the bottom center is a red note '4. Be sure to save!'. The bottom of the screen shows a navigation bar with page numbers '1 - 5 / 912' and arrows.

1. Choose the type of object you wish to translate from the **Object** drop-down menu, such as "Data elements".

- Be sure you have set the **Target Locale** to the correct language.
- 2.
 3. Choose the specific object you wish to translate, and translate each of the properties (Name, Short name, Description, etc). These properties vary from object to object.
 4. Press "Save" when you are done translating the specific object to save your changes.

Note

You can search for a specific term using the search feature in the upper right hand corner of the app.

DHIS 2 Documentation Guide

DHIS 2 Documentation System Overview

DHIS 2 is a web-based information management system under very active development with typically three releases per year. Each release typically includes a number of new features and additional functionality. Given the fast pace of development, the system's wide user base and distributed, global nature of development, a comprehensive documentation system is required.

In this chapter, we will describe the documentation system of DHIS 2 and how you can contribute.

Introduction

The DHIS 2 documentation is written in [Commonmark](#) markdown format. One of the main advantages of markdown is that there is complete separation between the content and presentation.

Commonmark is a strongly defined, highly compatible specification of markdown. Since markdown can be transformed into a wide variety of formats (HTML, PDF, etc) and is a text-based format, it serves as an ideal format for documentation of the system.

There exist a wide range of text editors which can be used for the creation of markdown files. For Linux and Windows, [ghostwriter](#) is a nice option; it is free and supports side-by-side preview and custom style sheets.

TIP

If you have the option to apply custom css to your markdown editor, set it to
./src/commonmark/en/resources/css/dhis2.css to reflect the
html output style for DHIS2!

One of the key concepts to keep in mind when authoring documentation in markdown, or other presentation neutral formats, is that the **content** of the document should be considered first. The **presentation** of the document will take place in a separate transformation step, whereby the source text will be rendered into different formats, such as HTML and PDF. It is therefore important that the document is well organised and structured, with appropriate tags and structural elements being considered.

It is good practice to break your document in to various sections using the section headings. In this way, very complex chapters can be split into smaller, more manageable pieces. This concept is essentially the same as Microsoft Word or other word processing programs. The rendering process will automatically take care of numbering the sections for you when the document is produced.

Getting started with GitHub

The DHIS 2 documentation system is managed at [GitHub](#) in its own source code repository. GitHub is a collaborative platform that enables multiple people to work on software projects collaboratively. In order for this to be possible, a version control system is necessary in order to manage all the changes that multiple users may make. GitHub uses the *git* source control system. While it is beyond the scope of this document to describe the functionality of *git*, users who wish to create documentation will need to gain at least a basic understanding of how the system works. A basic guide is provided in the next section. The reader is referred to the [git manual](#) for further information.

In order to start adding or editing the documentation, you should first perform a checkout of the source code. If you do not already have a GitHub account, you will need to get one. This can be done [here](#). Once you register with GitHub, you will need to request access to the *dhis2-documenters* group if you wish to modify the source code of the documentation directly.

Login to GitHub, and then file an issue [here](#). Your request will need to be approved by the group administrators. Once you have been granted access to the group, you can commit changes to the documentation branch and send and receive notifications if you wish. Alternatively, you can clone the documentation into your own repository, commit your changes to your own fork, and request that your changes be merged with the source of the documentation with a pull request [here](#).

Getting the document source

In order to edit the documentation, you will need to download the source of the documentation to your computer. GitHub uses a version control system known as git . There are different methods for getting Git working on your system, depending on which operating system you are using. A good step-by-step guide for Microsoft operating systems can be viewed [here](#). Alternatively, if you are comfortable using the command line, you can download git from [this page](#) If you are using Linux, you will need to install git on your system through your package manager, or from source code. A very thorough reference for how git is used is available in a number of different formats [here](#).

Once you have installed git on your system, you will need to download the document source. Just follow this procedure:

1. Make sure you have git installed.
2. On Windows systems, visit <https://github.com/dhis2/dhis2-docs> and press "Clone in Desktop". If you are using the command line, just type `git clone git@github.com:dhis2/dhis2-docs.git`
3. The download process should start and all the documentation source files will be downloaded to the folder that you specified.
4. Once you have the source, be sure to create your own branch for editing. Simply execute `git checkout -b mybranch` where *mybranch* is the name of the branch you wish to create.

Editing the documentation

Once you have downloaded the source, you should have a series of folders inside of the repository directory. The documents are structured as follows:

```
<root>
  └── src
    └── commonmark
      └── en
        ├── dhis2_android_user_man_INDEX.md
        ├── dhis2_developer_manual_INDEX.md
        ├── dhis2_end_user_manual_INDEX.md
        ├── dhis2_implementation_guide_INDEX.md
        ├── dhis2_user_manual_en_INDEX.md
        └── user_stories_book_INDEX.md
      └── resources
        ├── css
        │   ├── dhis2.css
        │   └── dhis2_pdf.css
        └── images
            └── dhis2-logo-rgb-negative.png
    └── content
      ├── android
      │   └── resources
      │       └── images
      ├── common
      │   └── bookinfo.yaml
      ├── developer
      │   └── resources
```

```

|   |   └ images
|   └ implementation
|       └ resources
|           |   └ images
|           └ resources
|               |   └ images
|   └ stories
|       └ resources
|           |   └ images
└ user
    └ resources
        └ images

```

The *_INDEX.md files are the starting points for the master documents. They contain only !INCLUDE directives.

e.g. dhis2_android_user_man_INDEX.md:

```

!INCLUDE "content/common/about-this-guide.md"
!INCLUDE "content/android/configure-dhis2-programs-to-work-on-android-apps.md"
!INCLUDE "content/android/android-event-capture-app.md"
!INCLUDE "content/android/android-aggregate-data-capture-app.md"
!INCLUDE "content/android/android-tracker-capture-app.md"

```

The !INCLUDE directives point to the "chapters" that are used to make up the manual.

NOTE

the !INCLUDE directives are not part of pure commonmark format, but are used in pre-processing to build the master documents. The particular format here is the one supported by markdown-pp out of the box, but we could change it to another "include" format if desired.

It is perfectly valid to use !INCLUDE directives in the sub-documents too, but currently the documents are split up at chapter level only.

- For the sake of convention, place all chapters in a folder in the following format:

`./src/commonmark/en/content/XXXX/<chapter>.md`

where XXXX represents one of the thematic folders which are used to organize the documentation, and <chapter> is the filename referenced from the *_INDEX.md file.

Using images

Image resources should be included inside a folder structure beginning with resources/images/ relative to the current document. e.g. for the chapter content/android/android-event-capture-app.md, the images are somewhere under content/android/resources/images/ <rest-of-path>.

This is important because the resources/images string is used to identify images in the files. Images will be collected under resources/images/content/android/<rest-of-path> relative to the master document, when the files are pre-processed for generation. *The paths are partially reversed to ensure they remain unique when collecting images from multiple thematic folders.*

Section references

In order to provide fixed references within the document, we can set a fixed text string to be applied to any section. For our markdown docs this is done by adding a comment after the section heading in the form:

```
<!-- DHIS2-SECTION-ID:name_of_section -->
```

where `name_of_section` is replace with the id you wish to use.

For example:

```
## Validation { #webapi_validation }

To generate a data validation summary you can interact ...
```

Will set the section id of the level 2 heading **Validation** to "webapi_validation", which may then be referenced as "#webapi_validation" within the html file.

After the full html file is generated, it is post-processed and the first DHIS2-SECTION-ID after the start of the section is used as the section id.

Please follow the convention of lowercase letters and underscores, in order to create id's that are also valid as filenames when the html files are split.

Tables

As an extension to pure commonmark, we also support *GFM tables* (defined with pipes |), such as:

Table Type	Description
Commonmark (HTML)	Tables described in pure HTML
Github Flavour Markdown (GFM)	Tables described with pipes: easier to read/edit, but limited in complexity

which produces output like:

Table Type	Description
Commonmark (HTML)	Tables described in pure HTML
Github Flavour Markdown (GFM)	Tables described with pipes: easier to read/edit, but limited in complexity

For simple tables these are much more convenient for working with. They are limited to single lines of text (i.e. each row must be on a single line), but you can, for example use
 tags to create line breaks and effectively split up paragraphs within cells, if necessary. You can also continue to use HTML tables when you really need more complexity (but you can also consider whether there is a better way of presenting the data).

DHIS 2 Bibliography

Bibliographic references are currently not supported in the markdown version of DHIS 2 documentation.

Handling multilingual documentation

The DHIS 2 documentation has been translated into a number of different languages including French, Spanish and Portuguese. If you would like to create a translation of the documentation or contribute to one of the existing translations, please contact the DHIS 2 documentation team at the email provided at the end of this chapter.

Committing your changes back to GitHub

Once you have finished editing your document, you will need to commit your changes back to GitHub. Open up a command prompt on Windows or a shell on Linux, and navigate to the folder where you have placed your documentation. If you have added any new files or folders to your local repository, you will need to add them to the source tree with the `git add` command, followed by the folder or file name(s) that you have added. Be sure to include a descriptive comment with your commit.

```
git commit -m "Improved documentation on organisation unit imports with CSV."
```

Finally, you should push the changes back to the repository with `git push origin mybranch`, where "mybranch" is the name of the branch which you created when you checked out the document source or which you happen to be working on. In order to do this, you will need the necessary permissions to commit to the repository. When you have committed your changes, [you can issue a pull request](#) to have them merged with the master branch. Your changes will be reviewed by the core documentation team and tested to ensure they do not break the build, as well as reviewed for quality. As mentioned previously, you can also push your changes to your own GitHub repo, if you do not have access to the main repo, and submit a pull request for your changes to be merged.

If you have any questions, or cannot find that you can get started, just raise a question on our [development community of practice](#).

Markdown support and extensions

This section attempts to capture the markdown and extensions that are supported for DHIS 2 documentation, and to provide a preview of the applied styles.

h3 Heading

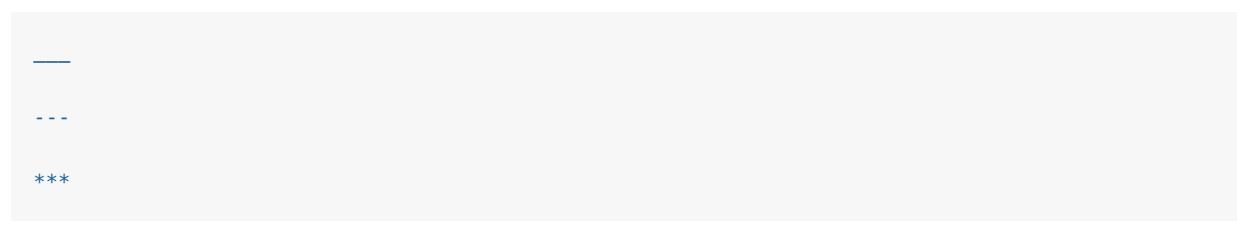
h4 Heading

h5 Heading

h6 Heading

Body text.

Horizontal Rules



Emphasis

This is bold text

This is bold text

This is italic text

This is italic text

Strikethrough

Blockquotes

Examples

Blockquotes can also be nested...

...by using additional greater-than signs right next to each other...

...or with spaces between arrows.

Markdown

```
> Blockquotes can also be nested...
>> ...by using additional greater-than signs right next to each other...
> > > ...or with spaces between arrows.
```

Code

Inline code

Inline highlighted code `var test = 0;` made within

```
`#!js var test = 0;`
```

Indented code

```
// Some comments
line 1 of code
line 2 of code
line 3 of code
```

Block code "fences"

```
Sample text here...
```

Syntax highlighting

```
var foo = function (bar) {
    return bar++;
};

console.log(foo(5));
```

Long lines

Buffalo buffalo (the animals called "buffalo" from the city of Buffalo) [that] Buffalo buffalo buffalo (that the animals from the city bully) buffalo Buffalo buffalo (are bullying these animals from that city).

Lists

Unordered

- Create a list by starting a line with +, -, or *
- Sub-lists are made by indenting 2 spaces:
 - Marker character change forces new list start:
 - Ac tristique libero volutpat at
 - Facilisis in pretium nisl aliquet
 - Nulla volutpat aliquam velit
- Very easy!

an indented code block

Ordered

1. Lorem ipsum dolor sit amet
2. Consectetur adipiscing elit
3. Integer molestie lorem at massa
4. You can use sequential numbers...
5. ...or keep all the numbers as 1.

Start numbering with offset?:

1. foo
2. bar

Multi-Levels

1. firstitem
2. still first level
 1. second level
 2. still second
 1. third level

```
a block of code at the third level
```

2. still third level
3. second level again
3. back to the first level
1. second
1. Oh, do stop it!

Tables

Return Parameter	Description
aoc	Attribute option combination identifier
pe	Period identifier
ou	Organisation Unit identifier
permissions	The permissions: 'mayApprove', 'mayUnapprove', 'mayAccept', 'mayUnaccept', and 'mayReadData' (same definitions as for get single approval status.)
state	One of the data approval states (same as for get single approval status.)
wf	Data approval workflow identifier

Right aligned and centred columns

Return Parameter	Description
aoc	Attribute option combination identifier
pe	Period identifier
ou	Organisation Unit identifier
permissions	The permissions: 'mayApprove', 'mayUnapprove', 'mayAccept', 'mayUnaccept', and 'mayReadData' (same definitions as for get single approval status.)
state	One of the data approval states (same as for get single approval status.)
wf	Data approval workflow identifier

Links

[link text](#)

[link with title](#)

Autoconverted link <https://github.com/dhis2>

Images

Images are shown full size with a max-width of 100%

```

```



Adding a title automatically causes rendering as a figure. Inline classes and styles can be added in curly brackets.

```
![DHIS2 Screenshots](resources/images/dhis2_screenshots.jpg "The Title"){ .center width=50% }
```



DHIS2 Screenshots

Admonitions

The following admonitions are supported, with pre-defined styles, in addition to general blockquotes.

Note

Note

Note

A note contains additional information which should be considered or a reference to more information which may be helpful.

Markdown

```
> **Note**  
>  
> A note contains additional information which should be considered or a  
> reference to more information which may be helpful.
```

Tip

Tip

Tip

A tip can be a useful piece of advice, such as how to perform a particular task more efficiently.

Markdown

```
> **Tip**  
>  
> A tip can be a useful piece of advice, such as how to perform a  
> particular task more efficiently.
```

Important

Important

Important

Important information should not be ignored, and usually indicates something which is required by the application.

Markdown

```
> **Important**  
>  
> Important information should not be ignored, and usually indicates  
> something which is required by the application.
```

Caution

Caution

Caution

Information contained in these sections should be carefully considered, and if not heeded, could result in unexpected results in analysis, performance, or functionality.

Markdown

```
> **Caution**  
>  
> Information contained in these sections should be carefully  
> considered, and if not heeded, could result in unexpected results in  
> analysis, performance, or functionality.
```

Warning

Warning

Warning

Information contained in these sections, if not heeded, could result in permanent data loss or affect the overall usability of the system.

Markdown

```
> **Warning**  
>  
> Information contained in these sections, if not heeded, could result  
> in permanent data loss or affect the overall usability of the system.
```

Work in progress

Work In Progress



Work In Progress

Information contained in these sections, will indicate that these are issues or errors we are currently working on.

Markdown

```
> **Work In Progress**  
>  
> Information contained in these sections, will indicate that these are issues or errors we are  
currently working on.
```

Example

Example

Example

A way to bring special attention to examples.

Admonitions can include a code blocks

```

var foo = function (bar) {
    return bar++;
};

console.log(foo(5));

```

Markdown

```

> **Example**
>
> A way to bring special attention to examples.
>
> Admonitions can include a code blocks
>
> ``js
> var foo = function (bar) {
>     return bar++;
> };
>
> console.log(foo(5));
> ```

```

Miscellaneous

May be possible depending on plugins/extensions

Mathematical equations

Blocks must be enclosed in `\[. . . \]` or `\begin{...} . . . \end{...}`.

Inline blocks must be enclosed in ``#!math ...`` or `\(...\)`.

Equations

$$\text{Indicator} = \frac{\text{BcgVaccinationsUnder1Year}}{\text{TargetPopulationUnder1Year}} \times 100$$

$$3 < 4$$

$$\begin{aligned} p(v_i=1|\mathbf{h}) &= \sigma(\sum_j w_{ij}h_j + b_i) \\ p(h_j=1|\mathbf{v}) &= \sigma(\sum_i w_{ij}v_i + c_j) \end{aligned}$$

The wave equation for u is

$$\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u$$

where ∇^2 is the spatial Laplacian and c is constant.

This equation is inline.

The homomorphism is injective if and only if its kernel is only the singleton set , because otherwise with such that .

Markdown

```

\[

\text{Indicator} = \frac{\text{BcgVaccinationsUnder1Year}}{\text{TargetPopulationUnder1Year}} \times 100

\]

\[3 < 4\]

\begin{aligned}

```

```

p(v_i=1|\mathbf{h}) & = \sigma\left(\sum_j w_{ij}h_j + b_i\right) \\
p(h_j=1|\mathbf{v}) & = \sigma\left(\sum_i w_{ij}v_i + c_j\right)
\end{align}

```

The wave equation for $\langle u \rangle$ is

```

\begin{equation}
\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u
\end{equation}

```

where $\langle \nabla^2 \rangle$ is the spatial Laplacian and $\langle c \rangle$ is constant.

This equation `#!math p(x|y) = \frac{p(y|x)p(x)}{p(y)}` is inline.

The homomorphism `#!math f` is injective if and only if its kernel is only the singleton set `#!math e_G`, because otherwise `#!math \exists a,b \in G` with `#!math a \neq b` such that `#!math f(a)=f(b)`.

Typographic replacements

Markdown	Result
(tm)	™
(c)	(c)
(r)	®
c/o	%
+/-	±
-->	→
<--	←
<-->	↔
=/=	≠
1/4, etc.	¼, etc.
1st 2nd etc.	1 st 2 nd etc.

Subscript / Superscript

```

- 19th
- H~2~0

```

- 19th
- H₂O

++Inserted text++

Marked text

Footnotes

Footnote 1 link¹.

Footnote 2 link².

Inline footnote³[Text of inline footnote] definition.

Duplicated footnote reference².

Definition lists

Term 1 : Definition 1 with lazy continuation.

Term 2 with *inline markup* : Definition 2

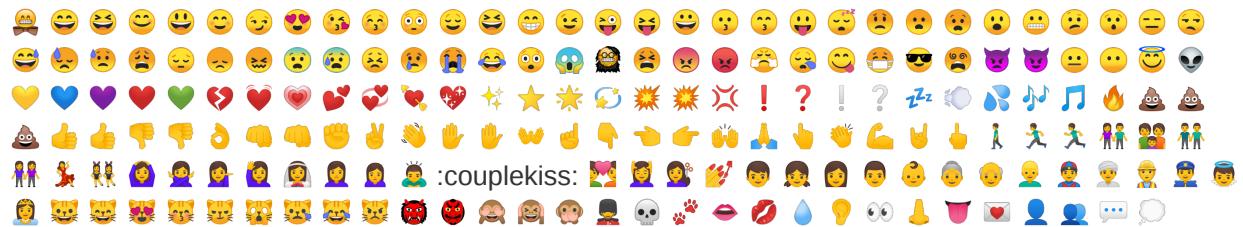
```
{ some code, part of Definition 2 }
```

Third paragraph of definition 2.

Emojis

Several sets of emojis are supported by entering the emoji name surrounded by colons: e.g :smile:. Below are list of common sets, those that are not rendered are currently not supported.

People



Nature



Objects



Places



Symbols





Submitting quick document fixes

For small changes to a document, it's actually very easy for anyone to submit changes without going through the whole process of raising a JIRA issue against DHIS 2. This can be done directly in [GitHub](#) and requires no knowledge of git. *All you need is a GitHub account!*.

This section is intended as a walk-through guide to making simple changes.

Typo fix walk-through

In this scenario, we are reading the documentation and find a typo (the word *manditory* should be *mandatory*):

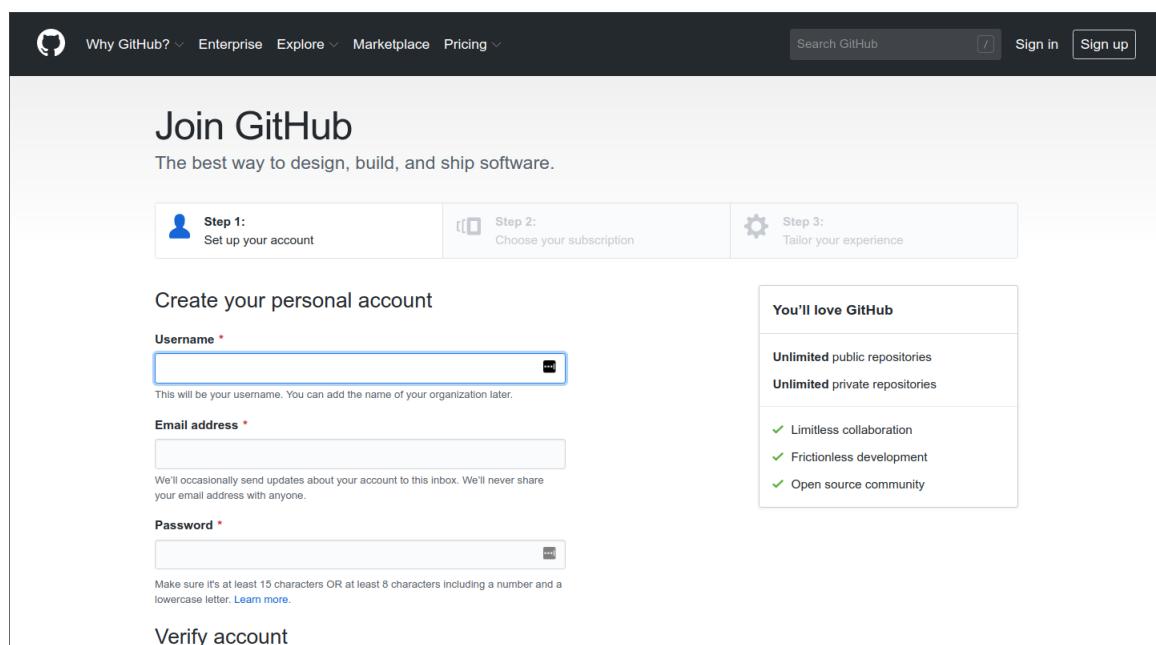
The screenshot shows a step-by-step guide for creating a tracked entity instance. Step 4 instructs to fill in attributes and enrollment details, and step 5 to click 'Create Tracked Entity Instance and Link'. A note below states: 'Note: When filling in data you might face a warning telling you that a possible duplicate has been found. You can click the warning to see these duplicates and if the duplicate is a match you can choose to link that Tracked Entity Instance by'.

A typo in Capture app documentation

This is in the chapter on "Using the Capture app" in the DHIS 2 User Manual. We want to fix this, so...

1. We log on to GitHub: <https://github.com/dhis2/dhis2-docs>

If you don't have an account, follow the Sign up link on the GitHub site
(public accounts are free)



GitHub sign in

2. Now we need to find the chapter that we want to change in GitHub. We simply go to the page we want to change on the DHIS2 documentation site and scroll to the top of the page. On the right side, above the main content, is an **edit** (pencil) icon. For pages with multiple versions, we select the one we want before editing; here we have selected the **master** version.

The screenshot shows the DHIS2 Documentation website. The top navigation bar includes links for User, Implement, Develop, Manage, Topics, and a dropdown for 'DHIS core version master'. The main content area features a heading 'Using the Capture app' and a sub-section 'About the Capture app'. A note box contains text about the Capture app replacing the Event Capture app. The right sidebar has a 'Table of contents' section with various documentation links. Two red circles highlight specific elements: one on the 'DHIS core version master' dropdown and another on the edit icon in the page header.

Page on documentation site

3. Clicking the **edit** icon on the docs site takes us directly to the relevant file in GitHub.

The screenshot shows a GitHub repository page for 'dhis2 / dhis2-docs'. The file 'using-the-capture-app.md' is open in a code editor. The top right corner of the editor has a red circle highlighting the edit icon (pencil). The code editor shows the content of the Markdown file, which includes sections like 'Using the Capture app' and 'Register an event'.

Capture app chapter on GitHub

4. From here we can choose to edit the file (*pencil icon*). An edit panel is displayed:

Here we have found and highlighted the offending word!

```

101
102 - You are now presented with a form for registering a new **Tracked Entity Instance**. You can choose to either register with
103 or without a program.
104 If a program is selected, the new **Tracked Entity Instance** will be enrolled in said program. You can also change the
105 **Organisation unit** by removing the one that is automatically set and selecting a new one.
106 !register_new_Tracked_Entity_Instance](resources/images/capture_app/register_te1.png)
107 4. Fill in the desired (and possibly mandatory) attributes and enrollment details.
108 5. Click **Create Tracked Entity Instance and Link**.
109 > Note: When filling in data you might face a warning telling you that a possible duplicate has been found. You can click the
110 warning to see these
111 > duplicates and if the duplicate is a match you can choose to link that **Tracked Entity Instance** by clicking the **Link** button.
112 > If the warning is still present when you are done filling in data, you will not see the **Create Tracked Entity Instance
113 and Link** button.
114 > Instead you will be presented with a button called **Review duplicates**. When you click this button a list of possible
115 duplicates will be displayed. If any of these duplicates matches the **Tracked Entity Instance** you are trying to create you can click the
116 Link button, if not you can click the Save as new person button to register a new **Tracked Entity Instance**.

```

Edit

Don't worry about the blue warning at the top that says we don't have write access to the file!

We can make the change and preview it in the Preview changes tab if we want. Here is the preview:

4. Fill in the desired (and possibly **mandatory**) attributes and enrollment details.

5. Click **Create Tracked Entity Instance and Link**.

Note: When filling in data you might face a warning telling you that a possible duplicate has been found. You can click the warning to see these duplicates and if the duplicate is a match you can choose to link that **Tracked Entity Instance** by clicking the **Link** button. If the warning is still present when you are done filling in data, you will not see the **Create Tracked Entity Instance and Link** button. Instead you will be presented with a button called **Review duplicates**. When you click this button a list of possible duplicates will be displayed. If any of these duplicates matches the **Tracked Entity Instance** you are trying to create you can click the **Link** button, if not you can click the **Save as new person** button to register a new **Tracked Entity Instance**.

Edit an event

1. Open the **Capture** app.
2. Select an organisation unit.
3. Select a program.

Preview

5. Finally we can submit our change as a *Pull Request (PR)*. We add a title for the change (and an *optional* description) and click **Propose file change**

The screenshot shows a GitHub interface. At the top, there is a table with one row, showing the date '4/1/2018' and the status 'No'. Below the table, there is a note: 'Note: You can download an event list in JSON, XML or CSV formats.' In the center, there is a modal window titled 'Propose file change'. The modal has two tabs: 'Fix a typo' (which is selected) and 'Fix a typo (walk-through)' (which is highlighted with a blue border). At the bottom of the modal are two buttons: 'Propose file change' (green) and 'Cancel' (red). At the very bottom of the page, there is a footer with links: '© 2019 GitHub, Inc.', 'Terms', 'Privacy', 'Security', 'Status', 'Help', 'Contact GitHub', 'Pricing', 'API', 'Training', 'Blog', and 'About'.

Submit the change

- a. We see a summary of our changes and click Create pull request:

The screenshot shows a GitHub pull request comparison between two branches. The base repository is `dhis2/dhis2-docs` and the head repository is `absentmind/dhis2-docs`, comparing branch `patch-2`. The status bar indicates "Able to merge". The commit history shows one commit from user `absentmind` on Jul 29, 2019, titled "Fix a typo". The diff view highlights a change in file `src/commonmark/en/content/user/using-the-capture-app.md`. The change adds a note about download event lists. The commit message is "Fix a typo". The commit hash is `960eb25`.

Submit the change

6. All done!

A pull request is now in the system, which can be easily reviewed and accepted by the DHIS 2 team.

The screenshot shows a GitHub pull request page for the repository `dhis2 / dhis2-docs`. The pull request is titled "Fix a typo #193" and is from the user `absentmind`. It has 3 commits and 1 check. The status bar indicates "+2 -2" with 2 green and 2 red bars.

Review required: At least 1 approving review is required by reviewers with write access. [Learn more.](#)

Some checks haven't completed yet: 1 pending check. One check is labeled "Semantic Pull Request Pending — add a semantic commit or PR title".

Merging is blocked: Merging can be performed automatically with 1 approving review.

Leave a comment: A text area for comments with a "Comment" button at the bottom.

Footer: © 2019 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#) | [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

Submit the change

Once accepted, the change will appear in the next build of the documentation!

Using JIRA for DHIS2 issues

Sign up to JIRA - it's open to everyone!

1. Go to: <https://jira.dhis2.org>.
2. Create an account with your name and email address.

Report an issue

Tip

Uncertain whether something is a missing feature, a bug or deprecated?
We'd really appreciate that you ask on the developer list before reporting a
bug directly. Thanks!

1. Click **Create** in the top menu.
2. Select a **Project** from the list.
3. Select an **Issue Type**:
 - **Improvement** - if you'd like to tell us about something that could be better such as usability or design suggestions.
 - **New feature** - if you want to suggest a feature.
 - **Task** - if you've been asked to work on a DHIS2 task.
 - **Bug** - if you've found something that needs fixing.
 - **Epic** - if you'd like to submit an idea for a new DHIS2 area such as an app. Epic is used for issues more complex than new features.
4. Click **Create**.

Tip

To create several issues in one go, select **Create another**.

5. Fill out the issue form. Please give us plenty of context! Include server logs, JavaScript console logs, the DHIS2 version and the web browser you're using.

Issues ▾ Boards ▾ Create Search

Create Issue Configure Fields ▾

Project * DHIS 2 (DHIS2)

Issue Type * Bug ?

- Improvement
- Task
- New Feature
- Epic

Summary *

Component/s

Description

Start typing to get a list of possible matches or press down to select.

Style ▾ B I U A ^A ?

Priority Medium ?

Create another Create Cancel

Search for issues

Click **Issues > Search for issues**.

If you click **Advanced**, you can order your search criteria using the predefined fields. You can also enter search terms in the search field. See also: [Search syntax for text fields](#).

[Issues ▾](#)[Boards ▾](#)[Create](#)[Current search](#)[Search for issues](#)

RECENT ISSUES

-  DHIS2-284 Analytics Sends fals...
-  DHIS2-149 Searching and sortin...
-  DHIS2-283 Spanish Interface no...
-  DHIS2-182 Organisation unit an...
-  DHIS2-30 Remove section

[more...](#)[Import Issues from CSV](#)

FILTERS

[My open issues](#)[Reported by me](#)[GIS features](#)[Manage filters](#)

About filters

You can save your search results as filters to get back to them faster. A filter is very similar to a favorite. We've created filters for features intended for release numbers 2.26, 2.27 and 2.28 and for all open bugs.

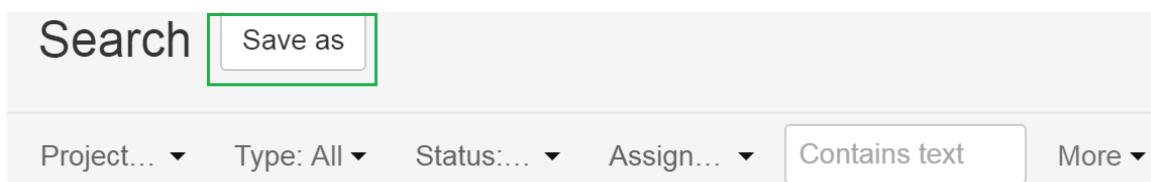
Create a filter

1. To create a filter, go to **Issues > Search for issues**.

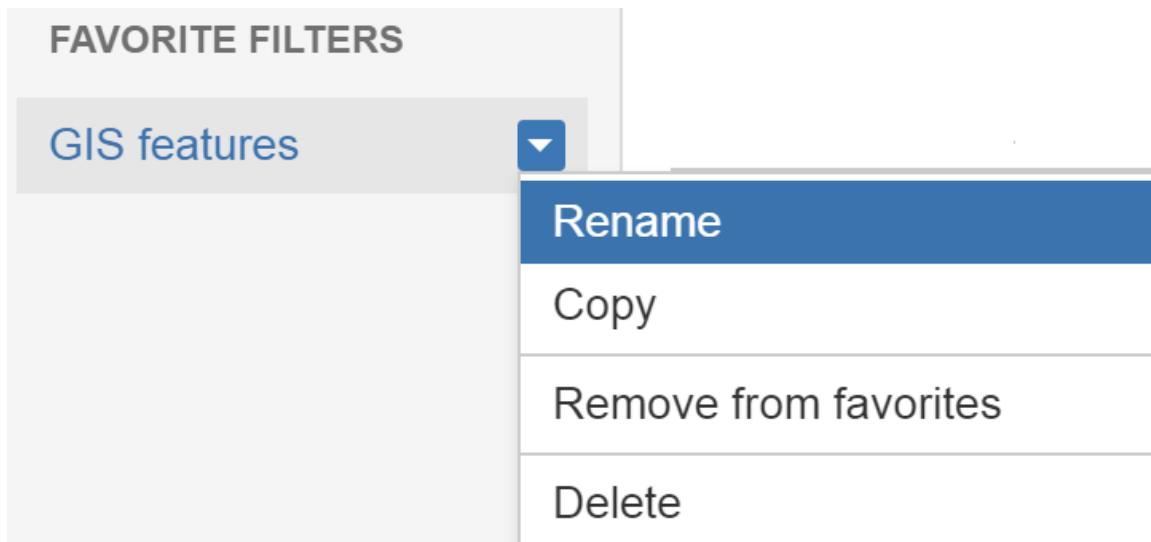
Add filters to your search such as:
2.

- **Project** – the main software project is DHIS 2.
- **Type** – you can filter by **Standard Issue Types** such as New Features, Improvements, Bugs or Epic or **Sub-Task Issue Types**.
- **Status** – filter by **To Do** (not started) and **Done**, for example.
- **Version** - click **More > All Criteria > Fix version** to filter by DHIS2 release number.
- **Internal**- click **More > Internal feature** to exclude low-level (back end) features from your search.

3. Click **Save As**. This button is at the top of the Search pane.



4. Enter a name for your search filter and click **Submit**. Your filter is now available in **Favorite Filters**. Use the arrow to modify your filter. Your filter is also available on the **System Dashboard**.



Add a filter to your profile

To add a filter to your profile, click **Issues > Manage filters** and click the star icon next to each filter.

Manage Filters

Favorite
My
Popular
Search

Favorite Filters

Filters are issue searches that have been saved for re-use. This page shows you all your favorite filters.

Name	Owner	Shared With	Subscriptions	
★ GIS features	test account (testaccount)	• Private filter	None - Subscribe	



Remove search filter terms from your search

Click the cross to remove search filter terms you previously added from your search.

The screenshot shows the JIRA search interface. At the top, there is a search bar labeled "Search" and a "Save as" button. Below the search bar are several dropdown filters: "Project... ▾", "Type: All ▾", "Status:... ▾", "Assign... ▾", "Contains text", "More ▾", "Fix Version: All ▾ (with a red X)", and "Internal feature: All ▾ (with a red X)". The "Fix Version" and "Internal feature" filters are highlighted with green boxes and have red X icons to their right, indicating they have been removed.

Communicate with us

To share information, clarify requirements, or discuss details about an issue, do this using issue comments.

1. Select the issue you want to comment.
2. In the Issue Detail view click **Comment** and enter your text.

To email others about your comment, simply enter **@User's Name** in the comment field. An email will be sent to the users' email addresses that are registered with their JIRA accounts.

3. Click **Add**.

Support

The DHIS2 community uses a set of collaboration and coordination platforms for information and provision of downloads, documentation, development, source code, functionality specifications, bug tracking. This chapter will describe these in more detail.

Home page: dhis2.org

The DHIS2 home page is found at <https://www.dhis2.org/>. The *Downloads* page provides links to downloads of the DHIS2 WAR files, the Android Capture mobile application, the source code, sample databases, and links to additional resources and guides. Please note that we provide maintenance patch updates for the last three versions of DHIS2. We recommend that you check back regularly on the download page and update your online server with the latest stable patch for your DHIS2 version. The version information and build revision can be found under the *About DHIS2* page inside your DHIS2 instance.

The navigation menu provides clear descriptions of the content of the site, and a search field in the top header allows you to easily search across the website.

Collaboration platform: community.dhis2.org

The primary DHIS2 collaboration platform is the *DHIS2 Community of Practice*. The site can be accessed at <https://community.dhis2.org/> and is based on the Discourse platform.

The Community of Practice is used to facilitate community support for DHIS2 user issues, as well as to help identify potential bugs in existing software versions and feature requests for future versions. It is also a place where members of the community can share stories, best practices, and challenges from their DHIS2 implementations, collaborate with others on projects, and offer their services to the larger community. Users can set up their Community of Practice accounts based on individual preferences for notification settings, and can reply to existing topics by email.

The *Support* section of the Community of Practice includes all topics that were created using DHIS2's previous collaboration platform, Launchpad, which is no longer active.

Bugs identified on the Community of Practice should be submitted to the DHIS2 core team on *Jira*.

Reporting a problem

If you encounter a problem while using DHIS2, take these steps first:

- Clear the web browser cache (also called history or browsing data) completely (you can use the Browser Cache Cleaner app in DHIS2; select all options before clearing).
- Clear the DHIS2 application cache: Go to Data administration -> Maintenance check "Clear application cache" and click "PERFORM MAINTENANCE".

If the problem persists, go to the Community of Practice and use key terms to search for topics that other users have posted which describe similar issues to find out if your issue has already been reported and resolved. If you are not able to find a thread with a similar issue, you should create a new topic in the *Support* category. Members of the community and the DHIS2 team will respond to attempt to assist with resolving your issue.

If the response you received in the Community of Practice indicates that you have identified a bug, you should post a bug report on the DHIS2 Jira.

Development tracking: jira.dhis2.org

Jira is the place to report issues, and to follow requirements, progress and roadmap for the DHIS2 software. The DHIS2 Jira site can be accessed at <https://jira.dhis2.org/>.

If you find a bug in DHIS2 you can report it on Jira by navigating to the DHIS2 Jira homepage, clicking *create* in the top menu, selecting "bug" as the issue type, and filling out the required fields.

For the developers to be able to help you need to provide as much useful information as possible:

- DHIS2 version: Look in the Help -> About page inside DHIS2 and provide the version and build revision.
- Web browser including version.
- Operating system including version.
- Servlet container / Tomcat log: Provide any output in the Tomcat log (typically catalina.out) related to your problem.
- Web browser console: In the Chrome web browser, click F12, then "Console", and look for exceptions related to your problem.
- Actions leading to the problem: Describe as clearly as possible which steps you take leading to the problem or exception.
- Problem description: Describe the problem clearly, why you think it is a problem and which behavior you would expect from the system.

Your bug report will be investigated by the Testing/QA team and be given a status. If valid its status will be set to "TO DO" and will be visible for the development team in their planning of milestones and releases. It can then be assigned to a developer and be fixed. Note that bugfixes are incorporated into the master branch and branches of up to the three latest (supported) DHIS2 releases - so more testing and feedback to the developer teams leads to higher quality of your software.

If you want to suggest new functionality to be implemented in DHIS2 you should first start a discussion on the Community of Practice to get feedback on your idea and confirm that the functionality you are suggesting does not already exist. Once you have completed these steps, you can submit a feature request on DHIS2 Jira by clicking "Create" in the top menu and selecting "Feature" as the issue type. Your feature request will be considered by the core development team and if accepted it will be assigned a developer and release version. DHIS2 users can vote to show support for feature requests that have been submitted. Existing feature requests can be browsed by using the "filter" function on Jira.

Source code: github.com/dhis2

The various source code branches including master and release branches can be browsed at <https://github.com/dhis2>

1. Footnote **can have markup**

and multiple paragraphs. ↵

2. Footnote text. ↵ ↵