[1. Data warehouse schema.](#_hj6famwkmgqk)

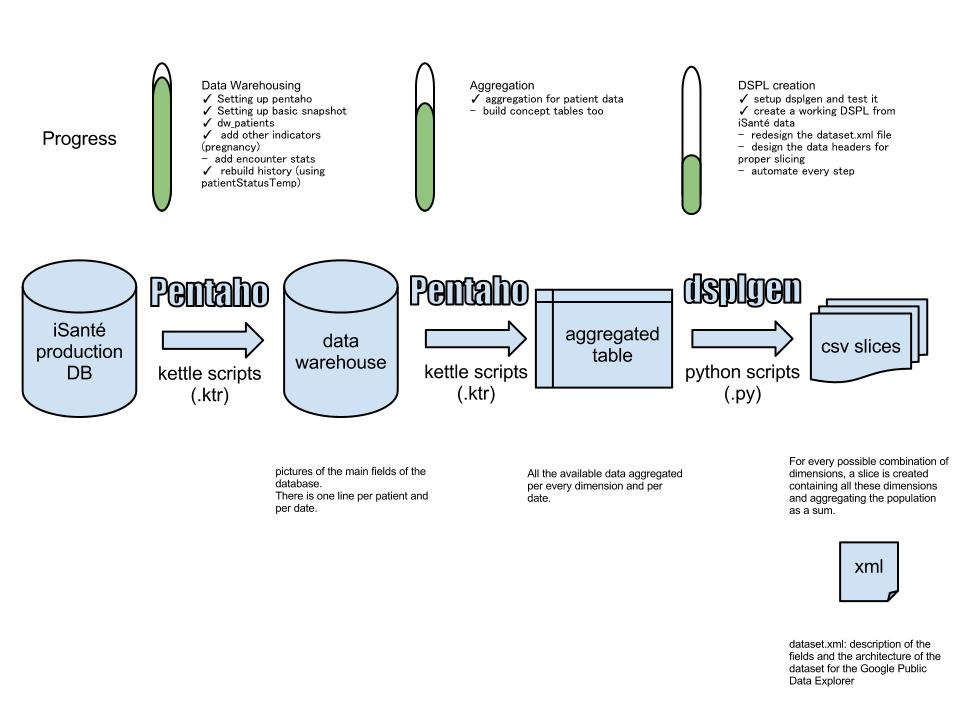
[2. Transformation/Computation/Aggregation.](#_7fixuuf7r4yk)

[3. Visualization.](#_w3g8lqfpvlsd)

The purpose of this document is to specify both the external and internal design of a generalized reporting mechanism suitable for TB, malaria, and other focused areas wherein data can be viewed both aggregated at the highest level (nationally, by department, by network) and also “drilled down” to display lower levels of aggregation, including the individual patient level.

There are three significant parts to this design:

1. data warehouse schema
2. transformation/computation/aggregation
3. visualization



# 1. Data warehouse schema.

In general, data warehouse schema will be significantly different from the iSanté transactional tables (except those that are already centered around a specific subject area such as vitals, and are fully denormalized) such as the obs table which has a key-value structure. Instead, data warehouse tables will generally be denormalized, organized by subject area, and keyed on patientid and visitdate. In the example of malaria, we might end up with a single table, defined something like this:

CREATE TABLE IF NOT EXISTS `dw\_malaria\_snapshot` (

`patientid` varchar(11) NOT NULL,

`visitdate` date NOT NULL default '0000-00-00',

`malariaDxA` tinyint(1) NOT NULL default '0',

`malariaDx` tinyint(1) NOT NULL default '0',

`malariaDxG` tinyint(1) NOT NULL default '0',

`malariaDxSuspectedA` tinyint(1) NOT NULL default '0',

`malariaDxSuspected` tinyint(1) NOT NULL default '0',

`malariaDxSuspectedG` tinyint(1) NOT NULL default '0',

`sym\_malariaLT` tinyint(1) NOT NULL default '0',

`sym\_malariaGT` tinyint(1) NOT NULL default '0',

`feverLess2` tinyint(1) NOT NULL default '0',

`feverGreat2` tinyint(1) NOT NULL default '0',

`chloroquine` tinyint(1) NOT NULL default '0',

`quinine` tinyint(1) NOT NULL default '0',

`primaquine` tinyint(1) NOT NULL default '0',

`convulsion` tinyint(1) NOT NULL default '0',

`lethargy` tinyint(1) NOT NULL default '0',

`hematuria` tinyint(1) NOT NULL default '0',

`ictere` tinyint(1) NOT NULL default '0',

`anemia` tinyint(1) NOT NULL default '0',

`anemiaA` tinyint(1) NOT NULL default '0',

`anemiaG` tinyint(1) NOT NULL default '0',

`hemoglobine` float NOT NULL default '5000',

`creatinine` float NOT NULL default '5000',

`glycemie` float NOT NULL default '5000',

`malariaTest` tinyint(1) NOT NULL default '0',

`malariaTestRapid` tinyint(1) NOT NULL default '0',

`rapidResultPositive` tinyint(1) NOT NULL default '0',

`rapidResultNegative` tinyint(1) NOT NULL default '0',

`smearResultPositive` tinyint(4) NOT NULL default '0',

`smearResultNegative` tinyint(4) NOT NULL default '0',

PRIMARY KEY (`visitdate`,`patientid`),

KEY `pidIndex` (`patientid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8

The data warehouse also contains aggregate tables based upon rollup (slice, in Google terms) of information in the base (i.e. dw\_malaria\_snapshot) tables. This looks like:

CREATE TABLE `dw\_malaria\_slices` (

`org\_unit` varchar(50) NOT NULL,

`org\_value` varchar(50) NOT NULL,

`indicator` smallint(10) NOT NULL,

`time\_period` varchar(10) NOT NULL,

`year` int(11) NOT NULL,

`period` smallint(6) NOT NULL,

`value` float DEFAULT '0'

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

In order to driill down to the patient level following the specific service area indicators, a patients table is also needed:

CREATE TABLE IF NOT EXISTS dw\_malaria\_patients (

indicator smallint(10) NOT NULL,

time\_period varchar(10) NOT NULL,

`year` int(11) NOT NULL,

period smallint(6) NOT NULL,

patientid varchar(11) NOT NULL,

PRIMARY KEY (indicator,time\_period,`year`,period)

) ENGINE=InnoDB DEFAULT CHARSET=utf8

To support display of indicators, existing and new lookup tables will be necessary. For example, the patient table and/or a data warehouse version of the patient table could be necessary. A pregnancy status table will most likely be necessary. The clinicLookup table will be necessary for geographic and network level aggregation. A lookup table is necessary for display of indicator definitions:

CREATE TABLE `dw\_malariaReportLookup` (

`indicator` smallint(6) NOT NULL,

`nameEn` varchar(100) NOT NULL,

`nameFr` varchar(100) NOT NULL,

`data\_elements` varchar(100) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

# 2. Transformation/Computation/Aggregation.

This refers to the process of converting transactional information in the iSanté database into persistent long-term historical data that can be directly used in visualization. In general, this will consist of reading normalized data and casting it into the de-normalized data warehouse format on an ongoing basis. The effort should attempt to minimize computations by adding only changed data based upon the lastModified key in the encounter record. The basic transformation would be something like this for obs data:

select date, patientid, name, value from obs where name in (‘suspectedMalaria’,

’confirmedMalaria’, ’treatmentMalariaDate’,etc) where lastModified > lastTransformation;

loop {

switch (name) {

case: confirmedMalaria

case: suspectedMalaria

etc.

}

insert into dw\_malaria\_snapshot (...) values (...) on duplicate key update … = ...;

}

and similar loops for other concepts such as lab data, but with more specific criteria. For example, here is an example query for labs related to malaria:

select e.visitdate, p.patientID,

case when p.testNameFr in ('Hemoglobine','Hémoglobine') then 'hemoglobine'

when p.testNameFr in ('creatinine', 'Créatinine') then 'creatinine'

when p.testNameFr in ('Glycemie', 'Glycémie') then 'glycemie'

else p.testNameFr end as testNameFr, p.result, p.result2, case

when (p.testNameFr = 'Malaria' and p.result = 1 and p.result3 like '%+%') or (p.labID between

1208 and 1211 and p.result <> 'Negatif' or p.result like '%+%') then 'smearResultPositive'

when (p.testNameFr = 'Malaria' and p.result = 2 and p.result3 like '%+%') or (p.labID between

1208 and 1211 and p.result = 'Negatif') then 'smearResultNegative'

when (p.testNameFr = 'Malaria' and p.result = 1 and p.result3 is null) or (p.testNameFr = 'Malaria

Test Rapide' and result = 'POS') then 'rapidResultPositive'

when (p.testNameFr = 'Malaria' and p.result = 2 and p.result3 is null) or (p.testNameFr = 'Malaria

Test Rapide' and result = 'NEG') then 'rapidResultNegative' else 'notMalariaTest' end as 'testType'

from a\_labs p, encounter\_snapshot e

where e.encounter\_id = p.encounter\_id and

e.dbSite = p.dbSite and p.result is not null and

(testnamefr in ('Hemoglobine','creatinine','Glycemie','Hémoglobine',

'Créatinine','Glycémie','Malaria','Malaria Test Rapide') or labID between 1208 and 1211) and

e.lastModified >= ?

And the rollups are computed:

/\* Indicator 12. Among suspected, percent (Rapid Test or Frottis Sanguin) / Suspected \*/

insert into dw\_malaria\_slices select '12', 'Week', year(visitdate), week(visitdate),

(sum(malariaTestRapid)+sum(malariaTest)+sum(rapidResultPositive)+sum(rapidResultNegative)

+ sum(smearResultPositive)+sum(smearResultNegative)) /

(sum(malariaDxSuspected)+sum(malariaDxSuspectedA)) from malaria\_snapshot where

(malariaDxSuspected = 1 or malariaDxSuspectedA = 1) and year(visitdate) = 2012 and

week(visitdate) in (43,44,45,46,47) group by 3,4;

/\* looping over all indicators (1-31) and all time period types (year,month,week) \*/

insert into dw\_malaria\_patients select $indicator, $period, $year, $period\_value, patientid from dw\_malaria\_snapshot where <indicator\_criteria> and year(visitdate) = $year and $period(visitdate) = $period\_value

# 3. Visualization.

Visualization is display of the data in the browser. Traditionally, this is done with a popup window that displays the data in a HTML table. The window additionally provides buttons for generating an Excel version as well.

We’ve developed a standard user interface for indicatord. Here’s an example (Malaria):

