# D4 – Challenge Fund Database Schema Updates – Final Report

*Paul Henshaw, GEM Foundation*
2019-05-03 v1.0

## Intended Audience

Large portions of this document are targeted at readers with database programming and/or administration skills and assume knowledge of general database constructs, the SQL language (particularly the PostgreSQL implementation), and geospatial operations (specifically PostGIS). We also assume familiarity with the Challenge Fund Round 2 Hazard, Exposure and Vulnerability databases and the concepts associated with natural hazards and risk assessment.

## Introduction

This document summarizes the activities undertaken as part of this project, namely the revision of existing database schemas and supporting software tools for hazard and exposure databases; the development of a new database schema and tools for losses; proposed changes to the MOVER vulnerability database; and population of the databases from the results of the GFDRR SWIO RAFI project.

We first describe the database schemas versions and the supporting software tools, outlining key changes compared to previous. We then describe the approach taken to import SWIO RAFI datasets into the various databases.

We conclude we a brief discussion of limitations and possible future directions including opportunities for continued collaboration with other initiatives such as the UK Space Agency METEOR project, and the work of the IDF/RMSG.

## Terminology

We use the PostgreSQL `SCHEMA` construct to implement namespaces. In order to avoid confusion, in this document we will use the term 'schema' to indicate the complete set of related database elements used to implement a database and the term 'namespace' to refer to the specific PostgreSQL implementation of a logical grouping of tables and related elements. All of the proposed schemas contain two namespaces: *cf_common* for common elements and a topic specific schema such as *ged4all*, *hazard* or *loss*.

## Database Schemas

In this section we present the final revised schemas for the Hazard and GED4ALL databases outlining the principal changes made to the versions described in deliverable D2. We also present the proposed changes to the MOVER vulnerability database schema to be shared and discussed with the Challenge Fund Round 2 partners University College London. Finally, we present the new draft schema for a database of the results of risk analyses, in particular for loss maps and loss curves.

The software and PostgreSQL schema definitions for the hazard, exposure and loss databases are available on GitHub under the terms of the open-source AGPL v3.0 license:

- https://github.com/gem/hazard_scenario_database/
- https://github.com/gem/ged4all
- https://github.com/gem/loss_database

The data imported into the databases is not made available via GitHub since the quantity of data makes this impractical.  We propose to provide with PostgreSQL database dump format files for the hazard, exposure and loss databases as a mechanism for transferring both the new schema and contents.

## Common Database Elements

As part of the database schema harmonization process, we make use of a *cf_common* namespace containing common tables, enumerated types for use any of the four Challenge Fund round 2 database schemas.

As described in document D2, we had initially planned to adopt the *hazard_enum* enumerated type defined by the MOVER vulnerability database however following discussion with Stuart Fraser of GFDRR, a more flexible approach using multiple tables was agreed.   This change was in part motivated by the somewhat inflexible nature of PostgreSQL enumerated types: it is easy to add new type values but is rather more difficult to rename or remove existing elements; however, the primary motivation was to facilitate interoperability between ThinkHazard! and other repositories of natural hazard and risk assessment data all of which currently use conceptually similar but incompatible classification systems.

Hazard types are validated via a traditional lookup-table approach using FOREIGN KEY constraints in order to guarantee referential integrity.   This allow us to constrain the permitted values associated with hazard and process types while also providing greater flexibility to change the categories in the future.

The *cf_common.hazard_type* table contains the following hazard codes and descriptions:

*Table 1: cf_common.hazard_type – valid hazard types*

| **code** | **name** |
|---|---|
| CF | Coastal Flood |
| CS | Convective Storm |
| DR | Drought |
| EQ | Earthquake |
| ET | Extreme Temperature |
| FL | Flood |
| LS | Landslide |
| MH | Multi-Hazard |
| TS | Tsunami |
| VO | Volcanic |
| WF | Wildfire |
| WI | Strong Wind |

Compared to the table presented in D2 there have been some minor changes to the names based on input from GFDRR but there have not been any conceptual changes.

In document D2 we noted that the Multi-Hazard category is present in the MOVER vulnerability database but did not anticipate or recommend storing hazard or loss data using this category. Following our experience importing loss data from the SWIO project, we find that we are obliged to revise our position since it appears to be not uncommon for both hazard and loss data to be aggregated across multiple perils.

Associated with each hazard category there are a number of process types which represent the physical phenomena causing the peril.
The *cf_common.process_type* table contains the process type codes and related information:

*Table 2: cf_common.process_type - hazard process types by category*

| hazard_code | code | name |
|---|---|---|
| CF | FCF | Coastal Flood |
| CF | FSS | Storm Surge |
| CS | TOR | Tornado |
| DR | DTA | Agricultural Drought |
| DR | DTH | Hydrological Drought |
| DR | DTM | Meteorological Drought |
| DR | DTS | Socio-economic Drought |
| EQ | Q1R | Primary Rupture |
| EQ | Q2R | Secondary Rupture |
| EQ | QGM | Ground Motion |
| EQ | QLI | Liquefaction |
| ET | ECD | Extreme cold |
| ET | EHT | Extreme heat |
| FL | FFF | Fluvial Flood |
| FL | FPF | Pluvial Flood |
| LS | LAV | Snow Avalanche |
| LS | LSL | Landslide (general) |
| TS | TSI | Tsunami |
| VO | VAF | Ashfall |
| VO | VBL | Ballistics |
| VO | VFH | Proximal hazards |
| VO | VLH | Lahar |
| VO | VLV | Lava |
| VO | VPF | Pyroclastic Flow |
| WF | WFI | Wildfire |
| WI | ETC | Extratropical cyclone |
| WI | TCY | Tropical cyclone |

As with the hazard types, there have been only cosmetic changes to the process types compared with the table proposed in D2.

The *cf_common.license* table provides a list of supported open-data licenses each with a unique id, short text code, full name, optional notes and a url for more detailed information.  This table is unchanged compared to the proposal outlined in document D2 and contains the following licenses:

- Creative Commons CCZero (CC0), https://creativecommons.org/publicdomain/zero/1.0/

- Open Data Commons Public Domain Dedication and Licence (PDDL), https://opendatacommons.org/licenses/pddl/index.html
- Creative Commons Attribution 4.0 (CC-BY-4.0), https://creativecommons.org/licenses/by/4.0/
- Open Data Commons Attribution License(ODC-BY), https://opendatacommons.org/licenses/by/summary/
- Creative Commons Attribution Share-Alike 4.0 (CC-BY-SA-4.0), http://creativecommons.org/licenses/by-sa/4.0/
- Open Data Commons Open Database License (ODbL), https://opendatacommons.org/licenses/odbl/summary/

The license table is referenced by the *contribution* tables contained in the specific databases. The structure of the *contribution* tables is identical for each schema, however the link to the corresponding hazard event set, exposure model, or loss model is specific to the schema so *contribution* is not present in the *cf_common* namespace. Additional licenses can be added to this table at any time via a simple INSERT statement, although care should be taken to ensure that changes are applied uniformly across the databases. Similarly changes to existing licenses can be performed via an UPDATE statement. Removing a license entry is slightly more complicated in that first any existing references to the license should be updated in order to maintain referential integrity.

The *cf_common.occupancy_enum* enumerated type proposed in D2 defines the valid categories of building occupancy/use for both the loss and exposure databases:

```
CREATE TYPE cf_common.occupancy_enum AS ENUM (
    'Residential',
    'Commercial',
    'Industrial',
    'Infrastructure',
    'Healthcare',
    'Educational',
    'Government',
    'Crop',
    'Livestock',
    'Forestry',
    'Mixed'
);
```

## Hazard Database Schema

In this section we first present a diagram of the changes to the hazard database schema and then highlight the key changes compared to the previous version.

The Entity-Relationship (ER) diagram below presents the revised hazard schema. There are no structural changes to the *hazard* schema compared to the version proposed in D2. As described above, there have been some minor changes to contents of the common hazard and process type tables.

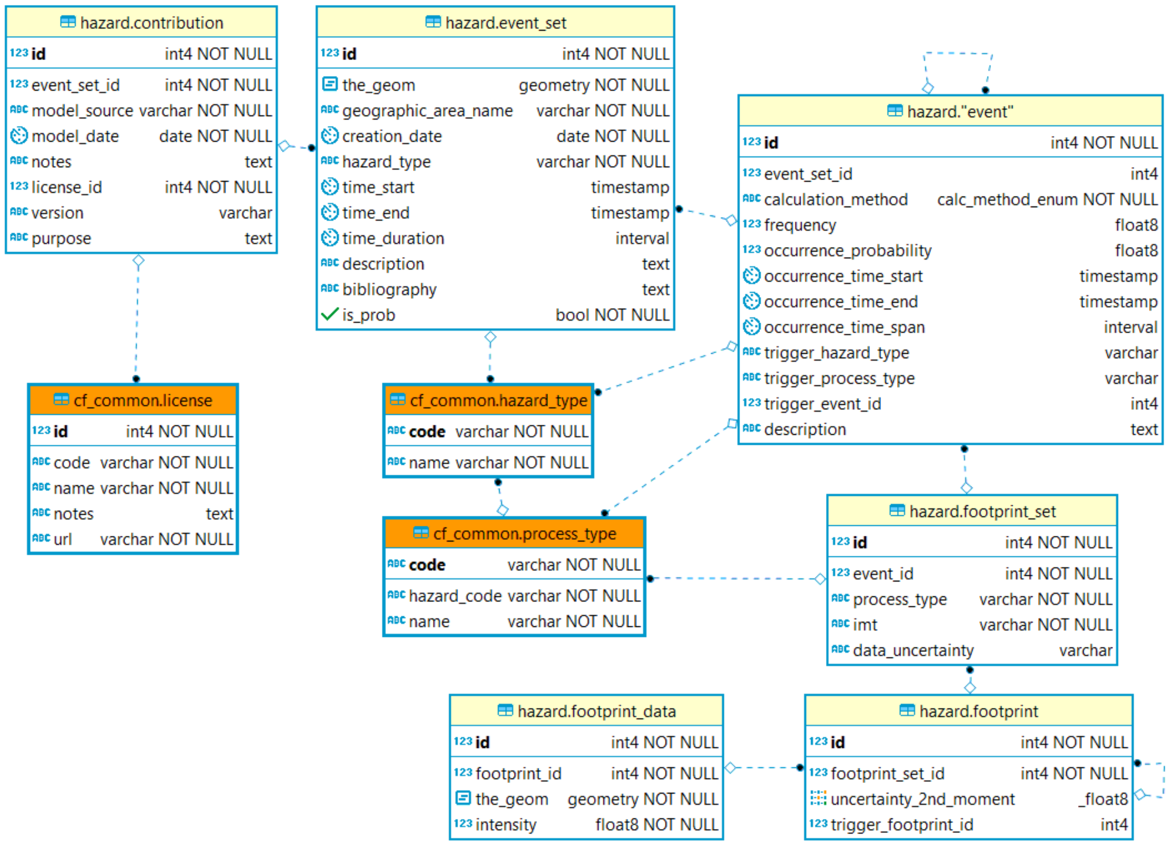


*Figure 1: Revised Hazard Schema ER Diagram*

In document D2 we described technical impediments to the implementation of a strict constraint for intensity type values in the *footprint_set.imt* field, due to the use of parameterized values such as ‘SA(0.3)’ (Spectral Acceleration with a period of 0.3 seconds). On further investigation, we have unfortunately not been able to devise a satisfactory solution to this problem at the database level and in addition have seen that there are also barriers to an at a conceptual level for some intensity types; please see the section on limitations below. For the moment, we have chosen to leave the *imt* field as a `VARCHAR` string and propose to continue discussions regard possible approaches with GFDRR.

## Hazard Import / Export Tools

During the initial Challenge Fund project, a Python scripts were developed to read hazard data into the database and to export data from the database into JSON format. Obviously, following changes to the database schema it was necessary to update these tools in line with the schema changes.

### Bulk Data Ingestion Support

While the import script worked correctly, one important limitation noted at the end of the original project was that larger datasets, such as the Volcanic ash-fall scenario for Rungwe, containing around 60 million data points, took a rather long time to load data into the database, in this case around 48 hours. The main reason for this speed limitation is that the Python script handles point data one row at a time. In the final report we identified the use of bulk-data ingestion techniques such as the PostgreSQL COPY command as a possible optimization strategy.

One of the objectives of this project is to ingest hazard, exposure and loss information produced by the WB SWIO project into the relevant CF databases. While analysing the available data we noted that a number of the hazard datasets were provided in the CSV format and that we might make use of the PostgreSQL COPY command to load these data files directly into a temporary table in the database. We then developed a new Python script to populate the CF schema table from meta-data descriptions using the JSON format defined in the previous project. By extending the JSON format to include a directive containing the part of a SQL query required to find the relevant subset of the previously loaded data, we were able to make use of a query of the form:

```
      INSERT INTO hazard.footprint_data
            (footprint_id, the_geom, intensity)
            (SELECT sub-query)
```

So that the sub-query extracts the sub-set of the pre-loaded data relevant for the given footprint and transfers it in a single bulk-data operation to the footprint_data table from the temporary table.

To illustrate the workflow, consider this example from the SWIO RAFI project results. The file “SWIO_ZAN_NTC_Flood_RP.csv” contains data describing Non-Tropical Cyclone Fluvial Flood hazard data of this form:

```
LocID,Lon,Lat,10-year,25-year,50-year,100-year,250-year,500-year,1000-year
1,39.65833,-4.84167,48.77,55.53,59.89,63.15,68.67,72.16,74.27
2,39.66667,-4.84167,55.46,63.25,68.22,72.04,78.32,82.34,84.66
3,39.675,-4.84167,59.3,67.74,73.07,77.26,84.04,88.43,90.79
4,39.68333,-4.84167,59.6,68.16,73.56,77.85,84.67,89.17,91.48
5,39.69167,-4.84167,57.39,65.72,70.94,75.17,81.82,86.22,88.47
...
```

We first create a temporary table (in this example we use a “temp” namespace) to store the data, grant access to the database user account we use to ingest data, and then use the COPY statement to ingest the CSV file into the temporary table:

```
CREATE TABLE temp.ntc_zan_fpf (
      LocID INTEGER NOT NULL,
      Lon DOUBLE PRECISION NOT NULL,
      Lat DOUBLE PRECISION NOT NULL,
      i10_year DOUBLE PRECISION NOT NULL,
      i25_year DOUBLE PRECISION NOT NULL,
      i50_year DOUBLE PRECISION NOT NULL,
      i100_year DOUBLE PRECISION NOT NULL,
      i250_year DOUBLE PRECISION NOT NULL,
      i500_year DOUBLE PRECISION NOT NULL,
      i1000_year DOUBLE PRECISION NOT NULL
);
GRANT SELECT ON temp.ntc_zan_fpf TO contributor;
COPY temp.ntc_zan_fpf FROM '/path/to/SWIO_ZAN_NTC_Flood_RP.csv'
      WITH (FORMAT CSV, HEADER);
```

While this example shows the use of the COPY command to read in CSV files, it is also possible for advanced users to use standard PostGIS tools such as shp2pgsql to read ESRI ShapeFile data into a temporary table.

Once the "raw" input data is available in a temporary table, we produce a JSON Meta-Data file to describe the hazard event set. Rather than including the footprint intensity values directly we can now specify a sub-query (in bold below) which extracts the relevant values from the temporary table:

```
"description": "SWIO RAFI Zanzibar Non-Tropical Cyclone Flood",
"creation_date": "2016-01-01",
"bibliography": "SWIO RAFI Report ...",
"hazard_type": "FL",
"geographic_area_name": "Zanzibar",
"is_prob": "True",
"events": [
{
  "calculation_method": "SIM",
  "description": "ZAN NTC Pluvial Flood 10yr event, depth (mm)",
  "frequency": 0.1,
  "footprint_sets": [
   {
    "imt": "Depth",
    "process_type": "FPF",
    "footprints": [
      {
        "_cf1_fp_data_query":
        "ST_SetSRID(ST_Point(lon,lat), 4326) AS the_geom,
           i10_year AS intensity FROM temp.ntc_zan_fpf"
      }
    ]
   }
  ]
}, …
```

Finally, we can import the data using the generic_scenarios.py Python script:

```
python generic_scenarios.py /path/to/meta-data.json
```

In this way the import execution time for a 17 million data-point dataset was reduced from several hours to less than five minutes. Complete JSON files for the SWIO meta-data and sub-queries are available in the hazard database GitHub repository:

https://github.com/gem/hazard_scenario_database/tree/master/data/swio

## GED4ALL Exposure Database Schema

The Entity-Relationship (ER) diagram below presents the revised GED4ALL schema:

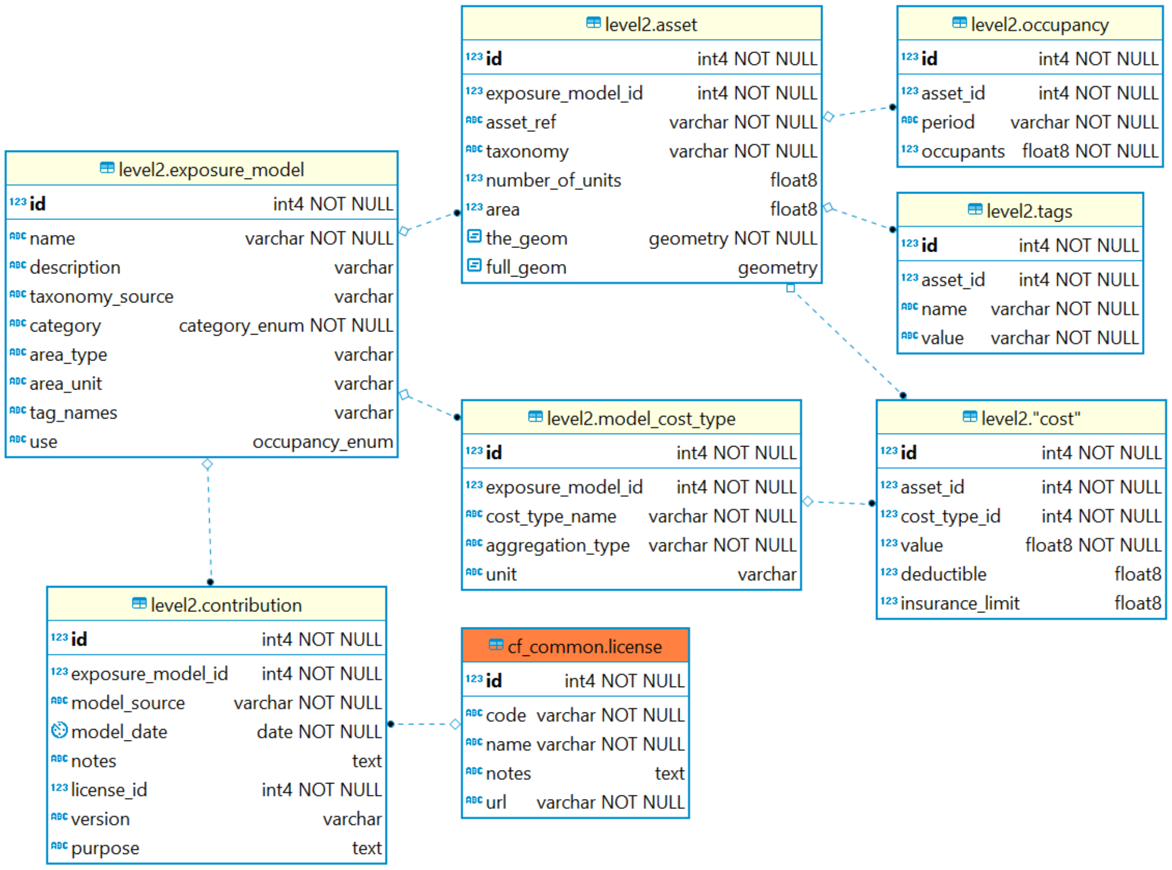


*Figure 2: Revised GED4ALL Exposure Schema ER Diagram*

The overall structure of the schema remains consistent with version presented in document D2. Note that the *cf_common* namespace contains the exact same elements as in the hazard table even though only the license table and *occupancy_enum* are used. Similarly, the *all_exposure* view continues to provide a convenient mechanism to extract presentation summaries, please see the Presentation Summaries section below for further details and examples.

The pre-existing import / export scripts have been revised to support the common database schema elements, however unlike the hazard and loss schemas, there is no direct support for bulk data ingestion. In order to maintain compatibility with the OpenQuake engine NRML format, these scripts make use of the underlying oq-common Python library which supports the CSV format for bulky exposure data but does not provide a mechanism for direct injection into a database. The performance of the ingestion script is now rather slower than the hazard or loss scripts, taking around 15 minutes to import around 900,000 rows of Madagascar gridded exposure data.

## MOVER Schema Updates

While we initiated informal discussions with UCL during the course of this project to inform them of the database harmonization activities, we have not been able to proceed to more concrete discussions regarding possible changes to the MOVER schema since the delivery of document D2. As described in D2, UCL have indicated that they are generally speaking supportive of the overall goals and would be prepared to consider updates to the MOVER schema however due to resource limitations they have a strong preference for doing so only once the changes have been applied and tested in the other databases. We propose to schedule a call with UCL following acceptance of the current schema, and in particular once the hazard and process categories described in the *cf_common* namespace are considered stable. We remain of the opinion that this hazard classification is an important step forward for Challenge Fund databases and that the MOVER database should also use the same approach. There is however, a not-insignificant difference in the list of intensity measure types present in the MOVER database compared to those discussed with GFDRR or with those used in the SWIO RAFI project.

As previously outlined in document D2, the MOVER *public.im_all_enum* enumerated type contains a number of intensity type values which are not used; we would certainly recommend removing these unless there is a compelling reason to keep them.

It would be desirable to include one or more database level constraint to ensure that the specified IMT is valid for the given hazard; however as described earlier, we have not been able to find an elegant solution for the hazard database and are not currently able to propose a solution for MOVER. Applying strict constraints which complicate or prevent contributors from sharing useful datasets is unlikely to further the goal of encouraging a standard unless there is critical mass of support for adopting a standard and tools available to facilitate conversion. We feel that there would be mutual benefit in the continued discussion and investigation of this topic including representatives of GFDRR, UCL, and possibly also other interested parties. It also possible that groups such as the IDF/RMSG might be interested in considering the topic of standardization of hazard and intensity types as part of their discussions regarding interoperability and compatibility.

## Loss Data Schema

In this section we present a loss database schema for storing loss maps and loss curves. The schema includes the common database elements described earlier along with a *contribution* table similar to those used in the Hazard and Exposure databases.

There are no significant conceptual changes to the loss schema compared to the version presented in document D2 however we have removed the *return_period* field and associated constraint checks from the *loss.loss_curve_map* table; having imported return period loss curves from the SWIO results, it has become apparent that this field was both redundant and misleading since the relevant return periods are stored in the *loss_curve_map_values.rates* array.

Another important change is the addition of the *all_loss_map_values* view which is intended to facilitate the generation of presentation summaries in a similar way to the GED4ALL *all_exposure* view. Please refer to document D2 for a more detailed presentation of the tables.

The Entity-Relationship (ER) diagram below presents the revised loss database schema.

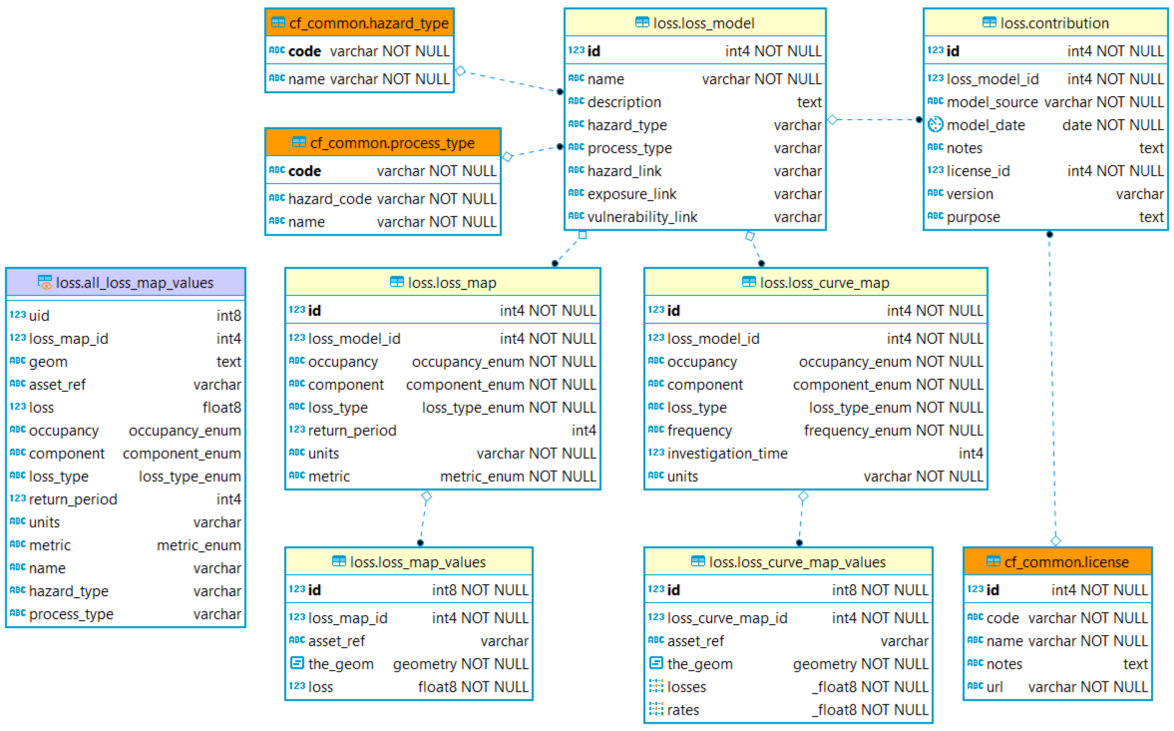


*Figure 3: New Loss Schema ER Diagram*

As with the hazard and exposure schemas, Python scripts have been developed to import and export data using the JSON format. The import script also supports bulk-data transfer using the same INSERT … SELECT sub-query approach adopted for the hazard database. Please refer to the GitHub repository for further details:

https://github.com/gem/loss_database/tree/master/python

## Presentation summaries, CSV and GeoPackage support

The database schemas were designed to store structured information across a number of different tables, which while flexible can be difficult for a non-expert to navigate. It is also not immediately obvious how the contents of the databases or a sub-set of them can be visualized as map layers in a desktop or web-based GIS system. In order to address this problem, the databases provide a means to produce "presentation summaries" which do not necessarily represent a complete view of the available data, are intended to be easier to use in a GIS environment.

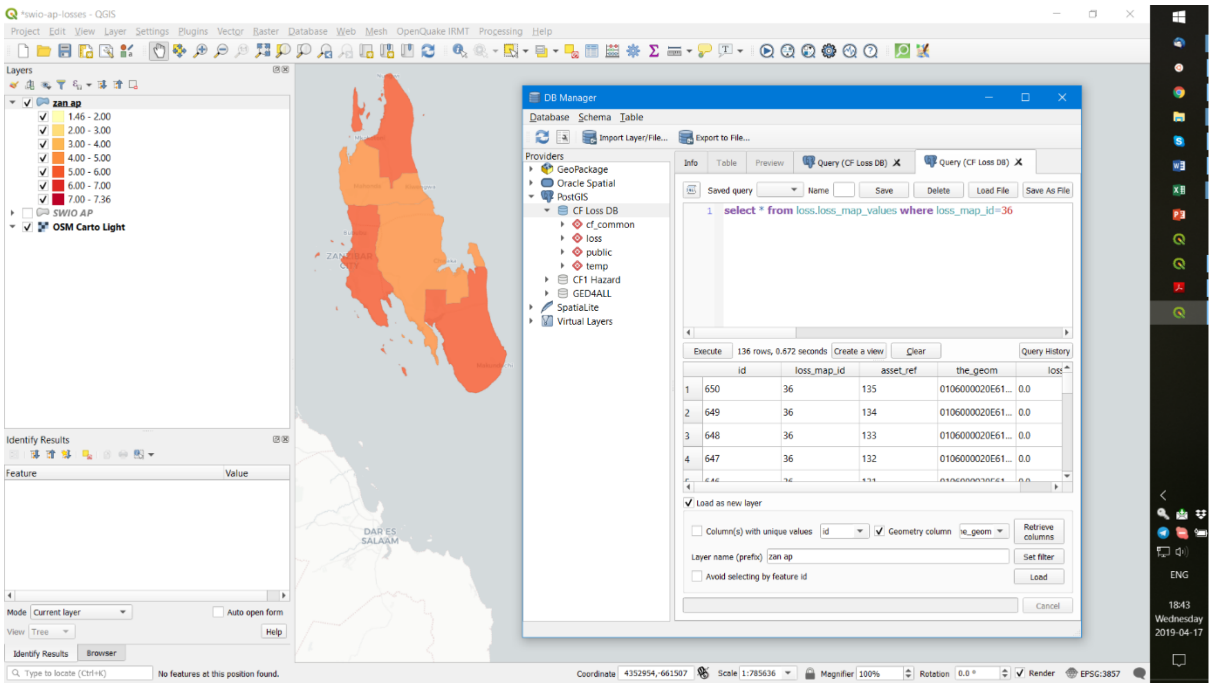
While there are differences in implementations of the various databases, the shared concept is for a view or stored procedure to return a subset of the data stored in the structured schema as a single, rectangular table, with a single geospatial location specification and meta-data associated with those values.

The GED4ALL *all_exposure* `VIEW` provide a convenient means to obtain a summary of the data suitable for presentation on a map and/or for exporting to CSV or GeoPackage format. The following SQL query returns all the assets and costs for the specified exposure model:

```
        SELECT * FROM level2.all_exposure WHERE exposure_model_id=136
```

The loss.loss_map_values view behaves in a similar way, the screenshot below illustrates the use of the QGIS DB Manager to execute the following query on the loss database:

```
        SELECT * FROM loss.loss_map_values WHERE loss_model_id=36
```



```
select * from loss.loss_map_values where loss_map_id=36
```

136 rows, 0.672 seconds

| | id | loss_map_id | asset_ref | the_geom | loss |
|---|---|---|---|---|---|
| 1 | 650 | 36 | 135 | 0106000020E61... | 0.0 |
| 2 | 649 | 36 | 134 | 0106000020E61... | 0.0 |
| 3 | 648 | 36 | 133 | 0106000020E61... | 0.0 |
| 4 | 647 | 36 | 132 | 0106000020E61... | 0.0 |

Load as new layer: ✓
Geometry column: the_geom
Layer name (prefix): zan ap

Figure 4: Loss Map using a simple query on the loss_map_values view

By wrapping a SELECT query in a COPY statement, we can export the results in CSV format:

```
        COPY (SELECT * FROM level2.all_exposure WHERE
               exposure_model_id=136)
        TO '/path/to/uganda-all-exposure.csv' WITH (FORMAT CSV, HEADER)
```

We were also asked to investigate the viability of using GeoPackage to distribute full models as part of this project. While we are convinced that GeoPackage is superior in every sense to the popular ESRI ShapeFile, and fully support its use as a “better ShapeFile” for GIS maps and layers, we are not in favour of using the GeoPackage format for arbitrary database structures.

The popular open-source GDAL package (https://www.gdal.org) offers a convenient means to generate GeoPackage files from the results of PostgreSQL/PostGIS query. The following command produces a GeoPackage for the visual summary of a single exposure model:

```
ogr2ogr -f GPKG uganda-all-exposure.gpkg \
      PG:'dbname=ged4all host=localhost user=ged4all_ro' \
      -sql \
      'SELECT *, SetSRID(ST_Point(lon,lat),4326) AS the_geom \
         FROM level2.all_exposure WHERE exposure_model_id=136'
```

When creating a GeoPackage it is important to ensure that the results of the SQL query contain a valid geometry column, the bold text in the query produces a valid Point geometry “the_geom” using the lat, lon columns produced by the GED4ALL *all_exposure* view.

While this is not a complete model and flattens the data into a single virtual table rather than following the GED4ALL schema, it produces a file which is clearly far easier to use with a desktop GIS or for use with web-based GIS systems such as GeoServer and GeoNode.

# Populating databases with SWIO RAFI results

The GFDRR SWIO RAFI project considered five regions in the South West Indian Ocean: Comoros, Madagascar, Mauritius, Seychelles, and Zanzibar (Tanzania). For each region, hazard exposure and loss data was produced along with reports, auxiliary datasets and maps. One of the tasks of this project was to import as much of the hazard, exposure and loss data as possible into the respective Challenge Fund databases.

## SWIO Hazard Data

During the execution of this project, we imported 25 SWIO RAFI hazard datasets as probabilistic event sets, 34 events for each country for a total of 170 hazard footprints and over 25million hazard footprint intensity values.

For Madagascar the following hazard events were imported, with a single footprint for each event. For most events a single footprint is provided for a single intensity measure, while for eathquakes two footprints are available for each event, one using PGA and a second in MMI.

*Table 3: SWIO Hazard Events for Madagascar*

| frequency | description | hazard_type | process_type | imt |
|---|---|---|---|---|
| 0.1 | MDG TC Pluvial Flood 10yr event, depth (mm) | FL | FPF | Depth mm |
| 0.04 | MDG TC Pluvial Flood 25yr event, depth (mm) | FL | FPF | Depth mm |
| 0.02 | MDG TC Pluvial Flood 50yr event, depth (mm) | FL | FPF | Depth mm |
| 0.01 | MDG TC Pluvial Flood 100yr event, depth (mm) | FL | FPF | Depth mm |
| 0.004 | MDG TC Pluvial Flood 250yr event, depth (mm) | FL | FPF | Depth mm |
| 0.002 | MDG TC Pluvial Flood 500yr event, depth (mm) | FL | FPF | Depth mm |
| 0.001 | MDG TC Pluvial Flood 1000yr event, depth (mm) | FL | FPF | Depth mm |
| 0.04 | MDG TC Storm Surge 25yr event, depth (m) | CF | FSS | Depth m |

| frequency | description | hazard_type | process_type | imt |
|---|---|---|---|---|
| 0.02 | MDG TC Storm Surge 50yr event, depth (m) | CF | FSS | Depth m |
| 0.01 | MDG TC Storm Surge 100yr event, depth (m) | CF | FSS | Depth m |
| 0.004 | MDG TC Storm Surge 250yr event, depth (m) | CF | FSS | Depth m |
| 0.002 | MDG TC Storm Surge 500yr event, depth (m) | CF | FSS | Depth m |
| 0.001 | MDG TC Storm Surge 1000yr event, depth (m) | CF | FSS | Depth m |
| 0.1 | MDG NTC Pluvial Flood 10yr event, depth (mm) | FL | FPF | Depth mm |
| 0.04 | MDG NTC Pluvial Flood 25yr event, depth (mm) | FL | FPF | Depth mm |
| 0.02 | MDG NTC Pluvial Flood 50yr event, depth (mm) | FL | FPF | Depth mm |
| 0.01 | MDG NTC Pluvial Flood 100yr event, depth (mm) | FL | FPF | Depth mm |
| 0.004 | MDG NTC Pluvial Flood 250yr event, depth (mm) | FL | FPF | Depth mm |
| 0.002 | MDG NTC Pluvial Flood 500yr event, depth (mm) | FL | FPF | Depth mm |
| 0.001 | MDG NTC Pluvial Flood 1000yr event, depth (mm) | FL | FPF | Depth mm |
| 0.1 | MDG EQ Shaking 10yr event | EQ | QGM | PGA |
| 0.1 | MDG EQ Shaking 10yr event | EQ | QGM | MMI |
| 0.04 | MDG EQ Shaking 25yr event | EQ | QGM | PGA |
| 0.04 | MDG EQ Shaking 25yr event | EQ | QGM | MMI |
| 0.02 | MDG EQ Shaking 50yr event | EQ | QGM | PGA |
| 0.02 | MDG EQ Shaking 50yr event | EQ | QGM | MMI |
| 0.01 | MDG EQ Shaking 100yr event | EQ | QGM | PGA |
| 0.01 | MDG EQ Shaking 100yr event | EQ | QGM | MMI |
| 0.004 | MDG EQ Shaking 250yr event | EQ | QGM | PGA |
| 0.004 | MDG EQ Shaking 250yr event | EQ | QGM | MMI |
| 0.002 | MDG EQ Shaking 500yr event | EQ | QGM | PGA |
| 0.002 | MDG EQ Shaking 500yr event | EQ | QGM | MMI |
| 0.0001 | MDG EQ Shaking 1000yr event | EQ | QGM | PGA |
| 0.0001 | MDG EQ Shaking 1000yr event | EQ | QGM | MMI |

Along with auxiliary datasets which are not hazard footprints and are not suitable for storage in the hazard database, the SWIO results also include a landslide susceptibility maps which we did not import since the values associated with the map were not hazard intensity but a Boolean indicator of whether the point was considered susceptible or not to landslides.

The screenshot below shows the result of running the following query via the QGIS DB manager to produce a single map combining all the SWIO earthquake shaking PGA intensity values for all five regions and for a 500yr return period (frequency = 1/500 = 0.002):

```
SELECT fd.id as fd_id, fd.the_geom, fd.intensity,
       es.hazard_type, fps.process_type, fps.imt, e.description
 FROM hazard.footprint_data fd
 JOIN hazard.footprint fp ON fp.id=fd.footprint_id
 JOIN hazard.footprint_set fps ON fps.id = fp.footprint_set_id
 JOIN hazard.event e ON e.id = fps.event_id
 JOIN hazard.event_set es ON es.id=e.event_set_id
WHERE es.description LIKE '%SWIO%' AND es.hazard_type='EQ' AND
       fps.imt='PGA' AND e.frequency=0.002
```

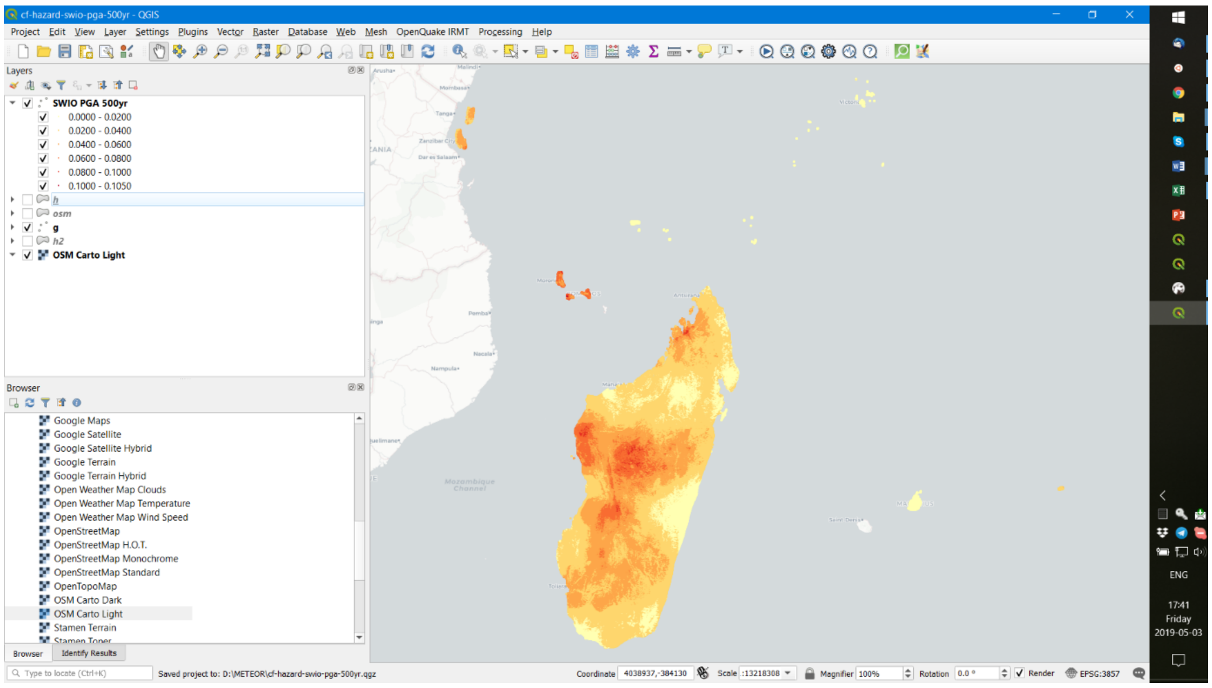


*Figure 5:SWIO Earthquake PGA map, 500yr*

## SWIO Exposure Data

We have been able to import some but not all of the SWIO gridded exposure data into the GED4ALL database. One important obstacle to this task is the lack of a mapping between the AIR occupancy and construction codes and the GED4ALL taxonomy. Our experience suggests that human intervention by domain expert is often required to map building typology codes from one taxonomy system to another, and that such mappings are not necessarily usable across projects and geographic areas.

The construction of a reliable AIR-GED4ALL taxonomy mapping represents a significant body of work and is outside the scope of this project. We are hopeful that activities undertaken in the context of groups such as the IDF/RMSG might facilitate cross-platform data exchange and the construction of conversion tools in the not too distant future.

In the meantime, we have imported some of the gridded exposure data, aggregated across all building and occupancy classes into GED4ALL using an “ALL” taxonomy code to demonstrate that the data can be stored, however the utility of the data is clearly limited since one can no longer distinguish between building typologies. One could also simply import the exposure data using the

AIR codes as a taxonomy string, however this approach is also of limited utility unless one already has vulnerability/fragility functions using the same taxonomy codes.

The screen shot below shows an example of an imported aggregate gridded exposure dataset.

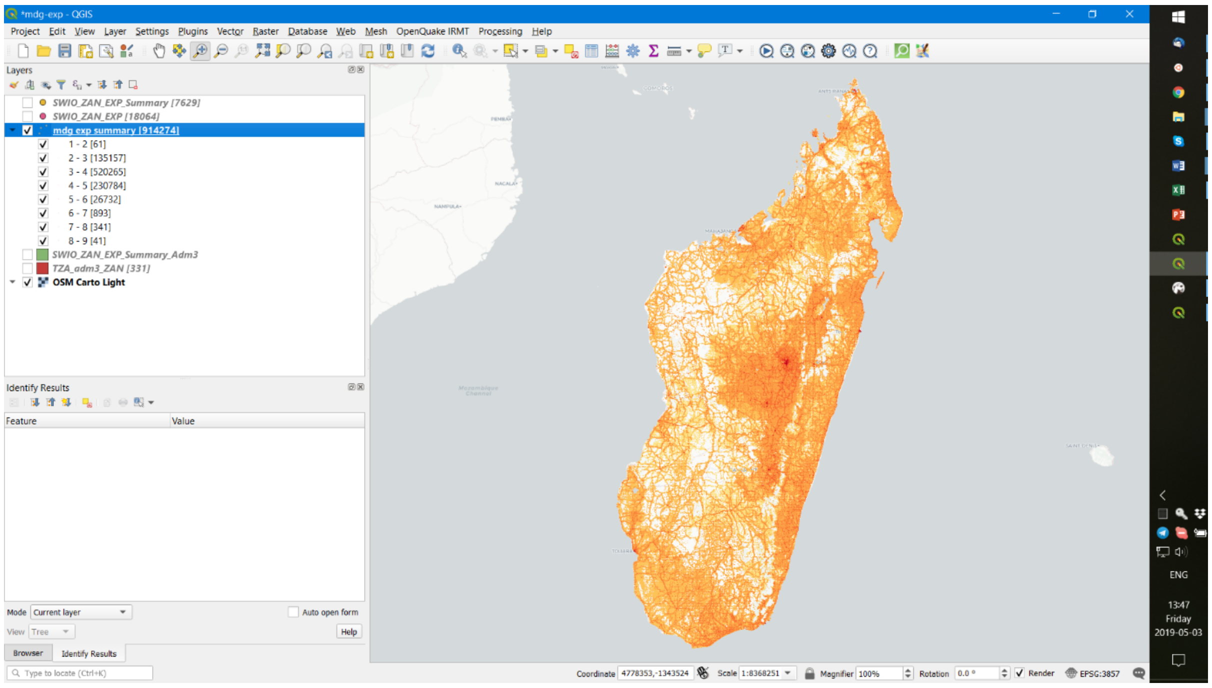


*Figure 6: SWIO Gridded Exposure map (aggregate USD values) for Madagascar*

## SWIO Loss Data

The results of the SWIO Project also include modelled losses for all five regions and for four categories of peril: All Perils (AP), Earthquake (EQ), Tropical Cyclone (TC), and Non-Tropical Cyclone (NTC). The earthquake loss data corresponds to our hazard/process categories of EQ and QGM respectively. For the other categories we have used a hazard type of MultiHazard (MH) and a NULL process type since multiple hazards are considered (in particular, both wind and flood for cyclones).

We have imported all twenty loss models, one for each regional and category of peril, each model containing a loss map of Average Annual losses and a loss curve map with a loss values for a range of seven return periods from 10 to 1000 years. Both loss map and loss curve maps are aggregated to the second administrative region.

The SWIO Loss data is more regular and consistent in terms of file format and naming conventions compared to the hazard data, so it was possible to automate some parts of the ingestion process. The scripts used to generate meta-data JSON descriptions using are available from GitHub:

https://github.com/gem/loss_database/tree/master/data/swio

The following screenshot shows a combined loss map for all perils and all SWIO regions:

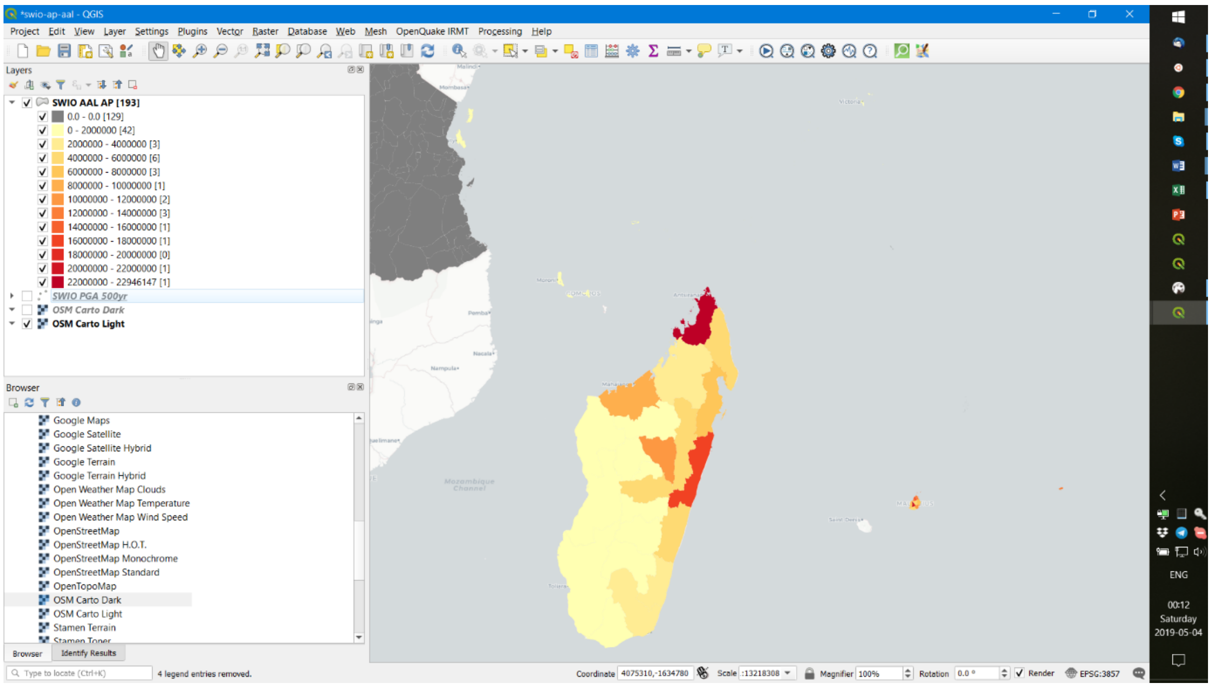
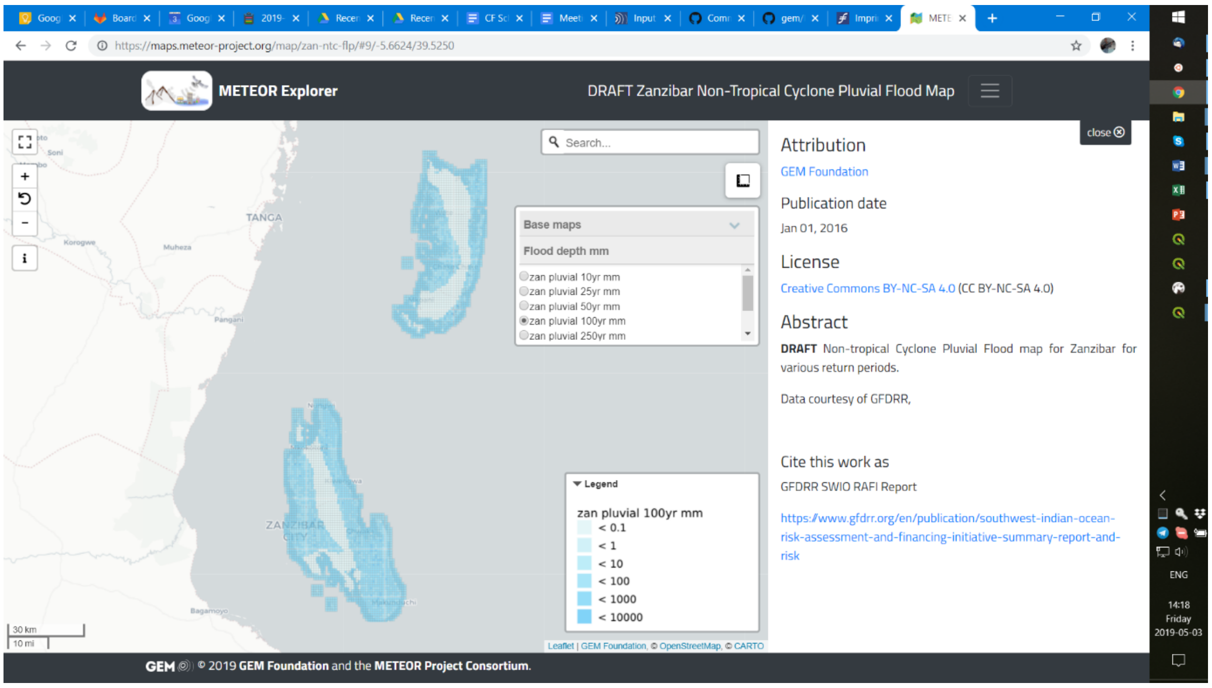


*Figure 7: Average Annual Losses for All Perils in all SWIO regions*

In the previous figure we can see that the administrative regions of continental Tanzania have been included in the loss map with 0 AAL values; this is probably the result of an oversight and is present also in the original SWIO loss data files.

## Links with other Initiatives

As already mentioned in previous documents, the Challenge Fund databases have attracted attention outside of the context of GFDRR projects and in particular the UK Space Agency funded, METEOR project, of which GEM is a partner has specifically included exchange of data with the Challenge Fund databases as a goal and activity.  METEOR has a specific focus on lower income areas with Tanzania and Nepal as pilot countries, so we have already been able to show that open-data produced in the original Challenge Fund projects exists and is relevant for the METEOR project. GFDRR has also agreed to share the SWIO data with the METEOR project under the terms of an open-license.  The following screenshot shows a preliminary flood map uploaded to the METEOR portal (currently available only to METEOR project partners) using SWIO Zanzibar pluvial flood data. As the METEOR project progresses, the partners also intend to share new datasets openly and, where possible include them in the appropriate Challenge Fund databases.  We feel that this bi-direction exchange of open-data is beneficial to both projects and the community in general and underlines the value of the open ethos in scientific collaboration.


DRAFT Zanzibar Non-Tropical Cyclone Pluvial Flood Map

Attribution

GEM Foundation

Publication date

Jan 01, 2016

License

Creative Commons BY-NC-SA 4.0 (CC BY-NC-SA 4.0)

Abstract

**DRAFT** Non-tropical Cyclone Pluvial Flood map for Zanzibar for various return periods.

Data courtesy of GFDRR,

Cite this work as

GFDRR SWIO RAFI Report

https://www.gfdrr.org/en/publication/southwest-indian-ocean-risk-assessment-and-financing-initiative-summary-report-and-risk

Base maps

Flood depth mm

zan pluvial 10yr mm
zan pluvial 25yr mm
zan pluvial 50yr mm
[✓ zan pluvial 100yr mm]
zan pluvial 250yr mm

Legend

zan pluvial 100yr mm

| | |
|---|---|
| ▢ | < 0.1 |
| ▢ | < 1 |
| 🟦 | < 10 |
| 🟦 | < 100 |
| 🟦 | < 1000 |
| 🟦 | < 10000 |

https://maps.meteor-project.org/map/zan-ntc-flp/#9/-5.6624/39.5250
METEOR Explorer
close
Search...
TANGA
Korogwe
Muheza
Pangani
ZANZIBAR CITY
Bagamoyo
30 km
10 mi
Leaflet | GEM Foundation, © OpenStreetMap, © CARTO
GEM © 2019 GEM Foundation and the METEOR Project Consortium.
ENG
14:18
Friday
2019-05-03

Figure 8: METEOR project portal showing preliminary flood map from SWIO data

## Limitations and Future Directions

In general, the schemas presented have demonstrated that they are capable of storing datasets useful for risk assessment and risk awareness. We have also provided tools to support import and export of existing datasets and have given some example queries illustrating how to use the databases via the QGIS DB Manager to visualize the contents. It is now possible to import large hazard datasets orders of magnitude more rapidly than before, and we have been able to apply the same techniques also to the new loss database. We are also satisfied that the hazard classification scheme discussed at length with Stuart Fraser of GFDRR has also proved fit for purpose for both the previously ingested datasets as well as the newly imported SWIO results.

We have, however, encountered some difficulties along the way which have been discussed in earlier sections of this document we feel it correct to summarize again here. The need for interoperability standards for risk data and building taxonomies in particular is neither new nor specific to the Challenge Fund database, however it has limited our ability to make the best use of the SWIO exposure data and is a topic that we believe is worthy of future discussion.

Another area for which additional discussion is needed relates to the use of strict validation constraints for process and intensity measures. In the Final Remarks of the D2 document, we stated:

> *In our opinion, the value of being able to validate hazard, process and IMT consistency values outweighs the risk of not being able to incorporate new datasets which use esoteric and/or newly developed techniques. Again, should we encounter datasets that do not fit the current approach during the ingestion phase, we will reach out to GFDRR to discuss further.*

We find ourselves now reconsidering this statement noting that the SWIO results are neither esoteric nor built on newly developed techniques, yet our previously planned approach was too rigid regarding support for aggregate losses and multiple perils.  Having relaxed some constraints to allow data ingestion we feel it appropriate to suggest that further discussion is necessary.