Deltares

Database volume analysis method

Database size analysis and recommendations

1207616-000

Deltares

Title
Database volume analysis method

Project	Reference	Pages
1207616-000	1207616-000-ZWS-0002	33

Keywords
Delft FEWS, Database Size

Summary
Database volume analysis method

References
Place references here

Version	Date	Author	Initials	Review	Initials	Approval	Initials

State
draft
This is a draft report, intended for discussion purposes only. No part of this report may be relied upon by either principals or third parties.

Contents

1 Introduction

Many systems are expanding over time. After the initial set up new models are added to the system or new (gridded) data feeds are introduced. All these changes have an impact on the size of the databases with the FEWS system. Finally, the introduction of ensemble forecasting can increase the size and complexity of the system further.

While the increase in coverage and complexity of the forecasting service is desirable, the associated increase in database size should be optimised where possible to ensure that the system performance remains acceptable. While increased database sizes do not pose technical problems when running the FEWS Software, such large databases do result in a loss of performance, in particular when synchronising data from the central server to distributed clients.

There are many factors that influence the size of the operational FEWS Databases. This includes the amount and resolution (spatial and temporal) of scalar and gridded data imported, the amount of data results from data manipulation and models runs within the system – and importantly the length of time data is kept within the database before it is purged by the rolling barrel process. In some cases data is kept that forms an intermediate result – and is not used in viewing results or in the production of a forecast product. This data could be set to temporary to avoid being stored at all. There are several other methods for optimising the size of operational FEWS Databases.

In this report an analysis of the sizing of the operational databases of The FEWS system is presented. An analysis of what constitutes the current size of the database is presented, and subsequently recommendations are made on how the database size could be improved. In the second chapter a brief outline of the methodology followed is provided. The third chapter provides the results of the analysis, with the fourth chapter providing recommendations on possible optimisations of the database size.

2 Methodology

2.1 Structure of the FEWS Database

To understand what constitutes the size of an operational FEWS Database it is good to have a basic understanding of the contents of the database. There are some 95 individual tables in the database. Not all of these are, however, important in determining the size of the database. Some of these tables contain data solely for administrative purposes. For the tables that are important, these can be broadly divided into two categories;

- Tables containing *static data*: These include mainly the tables with the configuration files. These are the XML and binary files used to configure the system, and subsequently loaded into the database. Under daily use these do not change, but multiple revisions will be kept in the databases which may lead to increasing database sizes.
- Tables containing *dynamic* data: They include mainly the time series data handled in the FEWS database. This is normally the largest volume of data, with the TimeSeries table containing all dynamic time series data (both scalar and grid). Other dynamic tables that are relevant include the WarmStates and the Archives tables.

2.2 Size of data in dynamic tables

The largest part if the FEWS database is normally taken by tables that include the *dynamic* data. The tables in this part of the database that will be considered in this analysis include;

- The *TimeSeries* table, which includes all time series data, including scalar and gridded data, ensemble data, import and simulation result data. Each time series in the database is a record, with the actual data values normally stored in the form of a binary object. Each record does have several additional fields that describe the record, which can be considered as overhead. Note that a single time series may be stored in multiple records, so where there are only a few values per record the overhead will be dominant, while for data such as gridded data the overheads will be small. The overheads vary little between the different records, while the size of the binary object with the data values will vary considerably. Normally the size of the database taken by the *TimeSeries* table is established using queries that count the size of the binary objects (which is reported in a separate field). In this analysis the size of the *TimeSeries* table as a whole will be considered, as well as the size of data from different sources (import data, simulation results, scalar data, gridded data etc.).
- The *WarmStates* table, which includes all model states stored in the database. Each record contains a single state from a single model (there will normally be multiple models per run). The size of the table is determined primarily by the size of the states, which are zipped and stored as a binary object. This table also includes overheads, but these will be small in comparison to the size of the state, unless the states are very small.
- The *Archives* and *ArchiveMetaData* tables include all archives made, and where these are made on a regular basis are considered a part of the dynamic database. Each record in the Archives table contains a single archive, which is again stored in the form of a binary object. The *ArchiveMetaData* tables includes a linked record, with a binary object containing the metadata to each archive. The overheads are small in this table when compared to the volume of the binary objects.
- The *TaskRunCompletions* table includes a record for each forecast task run. Each record includes an xml description of the task run and the configuration used. While the overall size of the table may not be large in terms of space on disk, a sizable table will result in a reduction of performance in synchronisation between databases.

There are additional tables with dynamic data in the database, but these are considered to be small in size in comparison to the other dynamic tables in the database.

The analysis of the database size will assess the size of these tables, as well as for the TimeSeries table how different types of data (gridded or scalar, import or simulation result) contribute to the overall size. The assessment is made using a combination of tools within the FEWS User Interface, as well as more detailed analyses using SQL statements run using the DBVisualiser database access software.

2.3 Size of data in static tables

While the dynamic tables may be significant in size, there are several static tables that will also contribute significantly to the overall size of the database on disk. These data contain primarily the XML configuration files, as well as in some cases ZIP files binary configuration files. Where such files are large and kept in multiple revisions, the size of the database may increase accordingly. As with the dynamic tables, there are only a few tables that contribute significantly and will be analysed. This includes the *ModuleInstanceConfigs*, the *ModuleDataSets* and the *ColdStates* tables. The size of the other configuration tables is considered small by comparison.

2.4 Scenarios for future increases of database size

Operational forecasting systems are never static, and as more detailed meteorological models and radar data become available, larger ensembles are used, and data networks increases, as well as more and more complex (gridded) models are used, the database sizes will increase accordingly.

Given an understanding of what contributes to the size of the database, estimating the impact of configuration changes can be easily estimated. Such impacts will come primarily from importing gridded data and running ensemble models, in particular models that produce gridded output data.

In this analysis, some of the future developments in The FEWS system will be assessed, and the impacts these may have on database sizes.

2.5 Recommendations on database size reduction

Following the analysis of what constitutes the size of the FEWS database, recommendations can be made to reduce this. All reductions will contribute to an improvement of system performance. There are different areas that can be considered in recommending reduction the amount of data kept;

- Reducing the expiry time for data. Perhaps the easiest method for reducing database sizes is to shorten the expiry time, which means data will be removed sooner from the database by the rolling barrel process. Typically expiry times differ depending on the type of data, and on the role of that data in the forecast process. When making recommendation on reducing the expiry times it is important to understand that role, and if the data is needed if it can be easily re-imported from an (external) archive or recreated from other data in the database.
- Removing Temporary Data. Most forecast workflows consist of several steps. Each workflow requires a set of input data and produces output data to be stored in the database. This output data may be either used in a subsequent workflow, a product, or in viewing results. If there are intermediate data between steps within a workflow, and these are not used in any subsequent step, do not appear in a product or are not available for viewing, then these need not be stored in the database. Setting these to temporary data will avoid the data becoming persistent – and result in database sizes being smaller.
- Removing or rescaling data not contributing directly to forecast and warning process. Many of the models, in particular distributed models produce results that are hydrologically interesting, but do not contribute directly to the forecast process. This may also depend if the data is a result in the historical run or the forecast run. If it does not contribute directly then it could either be made temporary to avoid it being stored, or rescaled to provide ancillary information at a lower resolution.
- Database optimisation. The FEWS database has some options to reduce the size of data stored. In the case of the time series table many records with only a few values will lead to unnecessarily large overheads, and can be reduced through running database maintenance scripts at regular intervals. Other options include the resolution at which parameters are stored.

2.6 Databases analysed

The operational FEWS system contains three databases that are closely related to each other. These are the Central Database (this is an ORACLE database, an SQL Server or Postgres), the Forecasting Shell Server database and the Operator Client database (these may be either Firebird or MS Access). The first of these two contain largely the same data, though the Central Database does contain additional administrative data used by the Master Controller, though that is limited in size. The Operator Client typically contains a sub-set of the data in the Central database, as determined by the synchronisation profile.

In this analysis only the Forecasting Shell Server and the Operator Client databases will be considered. The actual disk space of the database will depend on the vendor and how data is stored on disk (block sizes) – so there may be some differences between the values presented in this analysis and the actual disk space.

When analysing the database it is important to consider that the databases analysed are fully synchronised and that the system has been running under normal conditions for a length of time that exceeds the length of the rolling barrel processes. This is to ensure that the database size has reached a stable condition. The analysis will also consider the rate of data coming in – e.g. the amount of data from different processes in the space of a month. This is particularly relevant to the operator client, as the Forecasting Shelll Server database can be considered to contain all the data that the Central Database also has.

It is important when analysing the database through the FEWS User Interface using the options available in that to change the explorer configuration such that the rolling barrel is disabled.

3 Results

In this chapter the results of the analysis performed on example databases of a FEWS system are presented.

The results of the analyses are presented separately for the forecasting shell server and operator client databases.

3.1 Import data files

There are several sources of data that are imported into the FEWS system. While the size of these files is not directly related to the space that these data require in the database, this does give an impression on the amount of data imported, as well as the ratio between storage in the original files and in the database.

Table 1 provides an overview of the number and size of data files imported in the FEWS system on a daily basis. This data has been sampled based on a typical day (1st October 2012). Where file sizes are not constant these have been estimated based on the average of the files available on that day. While these sizes may vary from day to day, this table does show the order of the size of the data, with the gridded data formats (HYRAD and Met Office coastal data make up 95-97% of the volume of data imported.

Although this gridded data is kept in the system for a relatively short time, in some cases if a backlog of files spanning several or more days is imported in one go then this may result in a temporary spike in the database size. The size of these import files can later be compared to the size of the actual data imported on a daily basis.

Table 1 Overview of files imported in The FEWS system on a daily basis.

Import Source	**Type**	**Size of files (Bytes)**	**No Files/Day**	**Size/Day (MBytes)**
HYRAD	Radar Actuals	512928	288	140.88
	Radar Forecasts	16523	2302	36.27
	MOGREPS	62632	96	5.73
HIMS[1]	Telemetry	1845830	1	1.76
MetOffice	MOSurgeCS3	23451600	4	89.46
	MOWaveCS3	10109913	4	38.57
	MOWind	3435521	4	13.11
	MOWindAssim	649387.5	4	2.48
POL	Tidal Data	43558	96	3.99
SSE	Reservoir Data	2775	96	0.25
Total			**2895.00**	**332.50**

[1] *The HIMS data is provided to the system in multiple files per day at a frequency of an update very 10-15 minutes. As the file names are constant this file is constantly overwritten, and the estimated size is based on the final file of the day*

3.2 Total size of database tables

The Delft FEWS user interface provides a simple tool to view the size of the different tables in the database. This can be reached using the F12 option and selecting the *database* entry from the list, and *subsequently show time series disk space*. Table 2 provides an overview of all tables larger than 1MByte in both the FSS and OC databases. This shows that the time series table dominates the size. With the calculated size according to the FEWS log message of 6349 Mbytes for the FSS and 2048 MBytes for the OC databases these tables explain 98-99% of the size. It can be seen that several tables include significantly less data in the OC than in the FSS, which is due primarily to synchronisation profiles between the two. In some cases this means the table falls below the 1MB threshold (all tables feature in both databases), or that this data is not contained in the OC at all (e.g. WarmStates).

Table 2 Overview of tables larger than 1MByte disk space in the forecasting shell server database

Table Name	**No. Records FSS (-)**	**Size on Disk FSS (Mbytes)**	**No. Records OC (-)**	**Size on Disk OC (MBytes)**
TIMESERIES	274945	5939	153141	1741
MODULEINSTANCEDATASETS	39	201	39	201
REGIONCONFIGURATIONS	98	32	28	10
WARMSTATES	1580	28		
MODULEINSTANCECONFIGS	1254	20	1012	11
MAPLAYERS	36	19	32	19
LOGENTRIES	95784	15	73579	12
SYSTEMCONFIGURATIONS	28	9.5	10	1.5
REPORTS	265	5.4		
ARCHIVEMETADATA	164	3.6		
THRESHOLDEVENTS	14691	2.4		
SYSTEMACTIVITIES	85036	2.3		
DISPLAYCONFIGURATIONS	30	2.0		
TASKRUNCOMPLETIONS	164	1.6	166	1.6
TASKRUNS	15864	1.3	15877	1.3
MODULEINSTANCERUNS	16719	1.3	16735	1.3
REPORTIMAGES	52	1.1	52	1.1
Total		**6285**		**2001**

3.3 Structure of the TimeSeries table

Table 2 clearly shows that the *TimeSeries* table dominates the size of the databases for both Operator Client and Forecasting Shell Server. The structure of this table is explored further using a sequence of queries that explore what this table contains.

3.3.1 Relative size of different types of time series

The *TimeSeries* table contains a mixture of data from external and internal sources. Table 3 and Table 4 show the relative contribution to the time series table of the different types of data. In the FSS database it is clear to see that this is mainly historical data, while in the OC databases this is much less the case, with simulated forecasting data dominating. This may indicate there is redundant data in the historical run that is not synchronised to the OC. The smaller size of the OC database is again due to the synchronisation profiles. Note also that the total size of the table reported here is only that of the data itself as held in the binary objects, and not the complete table as reported above, While this difference can be attributed to overheads in the other fields, the rounding errors in reporting the database sizes make it difficult to estimate what the overhead per record is in terms of size on disk. The FSS database also includes a very small amount of temporary time series. This will happen if the amount of temporary data in a workflow is so large it exceeds the memory allocation. These are then flushed to the database during the run (ie become persistent). These are, however, not synchronised to the OC database en generally such volumes will remain small.

The data in these tables has been established using Query 1 in the Appendix.

Table 3 Distribution of the size of the TimeSeries table in different time series types (FSS Database)

Time series type	**No. Records (-)**	**Size on Binary Object (MBytes)**	**Average size of record (Bytes)**	**Percentage (%)**
External Historical	233679	2737	12283	47.25%
External Forecasting	10388	324	32688	5.59%
Simulated Historical	11116	1845	174004	31.84%
Simulated Forecasting	19654	868	46333	14.99%
Temporary	108	19	187374	0.33%
Total		**5793**	22095	100.00%

Table 4 Distribution of the size of the TimeSeries table in different time series types (OC Database)

Time series type	**No. Records (-)**	**Size on Binary Object (MBytes)**	**Average size of record (Bytes)**	**Percentage (%)**
External Historical	134735	304	2369	17.79%
External Forecasting	10072	265	27625	15.51%
Simulated Historical	3445	498	151515	29.09%
Simulated Forecasting	4889	644	138029	37.61%
Temporary	-	-	-	-
Total		1711	11716	100.00%

3.3.2 Relative size of gridded and scalar data types

Similar to the division in time series type, the ratio between different value types (scalar, longitudinal, gridded data) is shown in Table 5 and Table 6. This shows the strong dominance of gridded data in both databases.

The data in these tables has been established using Query 2 in the Appendix.

Table 5 Distribution of the size of the TimeSeries table in different value types (FSS Database)

Value type	**No. Records (-)**	**Size on Binary Object (MBytes)**	**Average size of record (Bytes)**	**Percentage (%)**
Scalar data	247663	157	664	2.71%
Gridded Data	27282	5637	216639	97.29%
Total		**5793**	22095	100.00%

Table 6 Distribution of the size of the TimeSeries table in different value types (OC Database)

Value type	**No. Records (-)**	**Size on Binary Object (MBytes)**	**Average size of record (Bytes)**	**Percentage (%)**
Scalar data	146533	82	584	4.77%
Gridded Data	6608	1629	258569	95.23%
Total		1711	11716	100.00%

3.3.3 Contribution of external historical data

In this paragraph the contribution of external historical data to the database size is established. This data constitutes to a large part data imported from external sources (for the historical domain), but also includes the results of selected data transformations such as correlations, rating curves and catchment averaging processes. The first column shows the percentage contribution of each module instance for total of the historical data, while the second percentage is relative to the total size of the time series table. Only the 10 largest contributing ModuleInstanceId's are shown.

The data in these tables has been established using Query 3 in the Appendix using ID=0.

Table 7 Distribution of the size data in the TimeSeries table according to module instance Id writing external historical data (FSS Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size ofdata Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
CatchmentAveraged_ObservedRainfall	Grid	11665	1966	71.84%	33.94%
HyradK_Convert_Units_Historical	Grid	1027	283	10.33%	4.88%
ImportNimrod	Grid	2959	281	10.28%	4.86%
ImportTelemetry	Scalar	117283	95	3.47%	1.64%
Interpolate_National_Historical	Grid	304	80	2.94%	1.39%
ImportMOTemperature	Grid	88	7	0.25%	0.12%
Astronomical	Scalar	9712	5	0.17%	0.08%
Performance_G2G_Forecasts	Scalar	11346	3	0.12%	0.06%
ImportAstronomical	Scalar	89	3	0.11%	0.05%
ResampleSSE	Scalar	4321	2	0.08%	0.04%

Table 8 Distribution of the size data in the TimeSeries table according to module instance Id writing external historical data (OC Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size of Binary Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
HyradK_Convert_Units_Historical	Grid	531	151	49.75%	8.85%
ImportTelemetry	Scalar	79009	48	15.60%	2.78%
ImportNimrod	Grid	579	45	14.91%	2.65%
Interpolate_National_Historical	Grid	133	35	11.64%	2.07%
ImportMOTemperature	Grid	90	7	2.39%	0.43%
ImportAstronomical	Scalar	108	4	1.35%	0.24%
Astronomical	Scalar	4662	4	1.30%	0.23%
Correlations_Tay	Scalar	2398	2	0.75%	0.13%
CatchmentAveraged_ObservedRainfall	Scalar	5455	2	0.66%	0.12%
ImportSSE	Scalar	2963	1	0.38%	0.07%

3.3.4 Contribution of external forecasting data

The query as used for determining the size of the external historical data can be applied to establish the contribution of external forecasting data. Table 9 and Table 10 provide an overview of the contribution of the different module instances, showing that in this case this is dominated by the gridded import data.

The data in these tables has been established using Query 3 in the Appendix using ID=1.

Table 9 Distribution of the size data in the TimeSeries table according to module instance Id writing external forecasting data (FSS Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size ofdata Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
ImportNWPExt	Grid	413	95	3.45%	1.63%
ImportMOTemperature	Grid	386	93	3.39%	1.60%
ImportMOSurge	Grid	176	43	1.57%	0.74%
ImportUKPP	Grid	133	32	1.17%	0.55%
ImportMOGREPS	Grid	116	24	0.87%	0.41%
ImportMOWind	Grid	60	16	0.60%	0.28%
ImportMOWave	Grid	37	9.23	0.34%	0.16%
Performance_PDM_Forecasts	Scalar	1842	5.83	0.21%	0.10%
Performance_ISIS_Forecasts	Scalar	1459	2.92	0.11%	0.05%
ImportMOSurge	Scalar	4683	1.53	0.06%	0.03%
ImportMOWind	Scalar	715	1.50	0.05%	0.03%
Performance_G2G_Forecasts	Scalar	217	0.18	0.01%	0.00%
ImportMOWave	Scalar	88	0.13	0.00%	0.00%

Table 10 Distribution of the size data in the TimeSeries table according to module instance Id writing external forecasting data (OC Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size of Binary Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
ImportMOTemperature	Grid	386	89	29.07%	5.17%
ImportMOSurge	Grid	248	60	19.77%	3.52%
ImportNWPExt	Grid	238	53	17.37%	3.09%
ImportMOWind	Grid	95	24	7.82%	1.39%
ImportMOGREPS	Grid	86	19	6.17%	1.10%
ImportMOWave	Grid	49	13	4.18%	0.74%
ImportUKPP	Grid	16	2.42	0.80%	0.14%
ImportMOSurge	Scalar	7106	2.32	0.76%	0.14%
ImportMOWind	Scalar	1073	2.28	0.75%	0.13%
Performance_PDM_Forecasts	Scalar	200	0.66	0.22%	0.04%
Performance_ISIS_Forecasts	Scalar	158	0.31	0.10%	0.02%
ImportMOWave	Scalar	165	0.24	0.08%	0.01%
Performance_G2G_Forecasts	Scalar	217	0.18	0.06%	0.01%
CatchmentAveraged_ForecastRainfall	Scalar	35	0.01	0.00%	0.00%

3.3.5 Contribution of simulated historical data

The query as used for determining the size of the external historical data is again applied to establish the contribution of simulated historical data. Table 11 and Table 12 show that in both FSS and OC database the largest volume of data is the gridded data, primarily from the G2G model, and a minor contribution from the Delft3D model. Note that only the 12 largest contributors of the total 334 module instances are shown.

The data in these tables has been established using Query 3 in the Appendix using ID=2.

Table 11 Distribution of the size data in the TimeSeries table according to module instance Id writing simulated historical data (FSS Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size ofdata Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
G2G_National_Pre_Historical_Interpolate	Grid	1932	604	32.74%	10.43%
G2G_National_InterpolateNWPExt	Grid	1201	324	17.59%	5.60%
G2G_National_Historical	Grid	1319	322	17.45%	5.56%
G2G_National_Post_Historical	Grid	634	177	9.60%	3.06%
G2G_National_MergeNWPTemp	Grid	594	169	9.15%	2.91%
G2G_National_Pre_Historical_Prep	Grid	434	120	6.50%	2.07%
G2G_National_Pre_Forecast_Interpolate	Grid	340	107	5.78%	1.84%
G2G_National_Post_Historical_Display	Grid	74	16.9	0.92%	0.29%
Delft3D_FoC_Historical	Grid	10	1.68	0.09%	0.03%
G2G_National_ToGauges_Historical	Scalar	620	0.94	0.05%	0.02%
Delft3D_FoC_Historical	Scalar	30	0.12	0.01%	0.00%
Tweed_FlowToLevel_Historical	Scalar	20	0.06	0.00%	0.00%

Table 12 Distribution of the size data in the TimeSeries table according to module instance Id writing simulated historical data (OC Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size of Binary Object (MBytes)**	**Percentage of type (%)**	**Percentage of total (%)**
G2G_National_Pre_Historical_Interpolate	Grid	506	154	30.91%	8.99%
G2G_National_Pre_Forecast_Interpolate	Grid	351	106	21.30%	6.20%
G2G_National_InterpolateNWPExt	Grid	327	88	17.73%	5.16%
G2G_National_MergeNWPTemp	Grid	188	53	10.66%	3.10%
G2G_National_Post_Historical	Grid	160	46	9.34%	2.72%
G2G_National_Pre_Historical_Prep	Grid	113	32	6.39%	1.86%
G2G_National_Historical	Grid	50	12	2.34%	0.68%
G2G_National_Post_Historical_Display	Grid	20	4.8	0.96%	0.28%
Delft3D_FoC_Historical	Grid	4	0.69	0.14%	0.04%
G2G_National_ToGauges_Historical	Scalar	155	0.24	0.05%	0.01%
Delft3D_FoC_Historical	Scalar	12	0.05	0.01%	0.00%
Tweed_FlowToLevel_Historical	Scalar	8	0.02	0.00%	0.00%

3.3.6 Contribution of simulated Forecasting data

The same query is again used but now to establish the contribution of simulated forecasting data. Table 13 and Table 14 show that as with the simulated historical data in both FSS and OC database the largest volume of data is the gridded data, primarily from the G2G model, and a minor contribution from the Delft3D model. Again only the 12 largest contributors of the total 345 module instances are shown.

The data in these tables has been established using Query 3 in the Appendix using ID=3.

Table 13 Distribution of the size data in the TimeSeries table according to module instance Id writing simulated forecasting data (FSS Database).

ModuleInstanceId	Value Type (-)	No. Records (-)	Size ofdata Object (MBytes)	Percentage of type (%)	Percentage of total (%)
G2G_National_Forecast_MOGREPS	Grid	1333	335	38.60%	5.79%
G2G_National_InterpolateNWPExt	Grid	903	243	27.95%	4.19%
G2G_National_Post_Forecast	Grid	340	94.1	10.83%	1.62%
G2G_National_Post_Forecast_MOGREPS	Grid	241	63.1	7.27%	1.09%
G2G_National_Post_Forecast_Display	Grid	139	38.5	4.43%	0.66%
G2G_National_Pre_Forecast_Prep	Grid	106	29.5	3.40%	0.51%
G2G_National_Forecast	Grid	97	24.0	2.76%	0.41%
Delft3D_FoC_Forecast	Grid	91	19.0	2.19%	0.33%
G2G_National_ToGauges_Forecast_MOGREPS	Scalar	169	5.37	0.62%	0.09%
G2G_National_Stats_Forecast_MOGREPS	Scalar	62	1.57	0.18%	0.03%
Delft3D_FoC_Forecast	Scalar	120	1.10	0.13%	0.02%
Coastal_ResampleReports_Forecast	Scalar	160	0.84	0.10%	0.01%

Table 14 Distribution of the size data in the TimeSeries table according to module instance Id writing simulated forecasting data (OC Database).

ModuleInstanceId	Value Type (-)	No. Records (-)	Size of Binary Object (MBytes)	Percentage of type (%)	Percentage of total (%)
G2G_National_Forecast_MOGREPS	Grid	1349	345	69.26%	20.15%
G2G_National_Post_Forecast	Grid	312	87	17.49%	5.09%
G2G_National_Post_Forecast_MOGREPS	Grid	254	67	13.37%	3.89%
G2G_National_InterpolateNWPExt	Grid	211	56	11.28%	3.28%
G2G_National_Post_Forecast_Display	Grid	110	31	6.15%	1.79%
G2G_National_Pre_Forecast_Prep	Grid	84	24	4.76%	1.39%
G2G_National_Forecast	Grid	85	21	4.32%	1.26%
G2G_National_ToGauges_Forecast_MOGREPS	Scalar	116	5.6	1.13%	0.33%
Delft3D_FoC_Forecast	Grid	17	3.2	0.64%	0.19%
G2G_National_Stats_Forecast_MOGREPS	Scalar	95	1.6	0.33%	0.10%
G2G_National_ToGauges_Forecast	Scalar	62	0.36	0.07%	0.02%
G2G_National_Post_Forecast_MOGREPS_Disp	Grid	4	0.35	0.07%	0.02%

3.3.7 Number of records with size of < 64 Bytes

A large number of very small records may result in a degradation of performance. For these small records the size of the binary time series data is small in comparison to the fields that describe the data. Table 15 shows the largest contributing module instances to small records. As expected these are all scalar time series, with the import processes such as import telemetry providing the largest contribution. The threshold of 64 bytes is perhaps somewhat arbitrary, but has been selected as this is taken as an estimate of the size of the overhead in each record, and thus when the balance tips from the data to the overhead being dominant per record.

Table 15 Number of records with less than 64 Bytes binary data (FSS Database).

ModuleInstanceId	**Value Type (-)**	**No. Records (-)**	**Size ofdata Object (Bytes)**	**Average size (Bytes)**
ImportTelemetry	Scalar	17528	482325	28
ImportPOL	Scalar	11929	258724	22
ImportMOSurge	Scalar	4321	149609	35
Astronomical	Scalar	3054	73135	24
CalculateMOSurgeHistorical	Scalar	1381	67669	49
Correlations_Flows	Scalar	2149	62321	29
Upper_FlowToFlow_Forecast	Scalar	2345	49067	21
CorrelationsArea1	Scalar	2345	49059	21
Correlation2	Scalar	2199	47469	22
Correlation1	Scalar	2173	46291	21
CatchmentAveraged_ObservedRainfall	Scalar	1270	37657	30
Correlations_Area2	Scalar	293	11389	39

3.3.8 Volume of data for specific workflows

Several workflows are run at regular intervals within The FEWS system. To understand the relative contribution of each to the amount of data in the database, the amount of data per workflow can be queried. There are two types of workflow. For a forecast workflow, the amount of data contributed will be relatively constant for each run, and an impression can be obtained by summing the data contributed by selected runs. For import and processing type workflows that contribute mainly to external data types, the amount contributed will depend on the amount of data available, and it is more informative to establish how much is contributed on a daily basis. The size of this data in the database can then be compared to the size of the source data files explored earlier.

Table 16 provided an overview of all the forecast task runs in the FSS database. Only those important to contributing data and being scheduled runs are shown. This shows the total size of data contributed by each workflow, the number of runs in the database, as well as the average size of the data contributed to each run. As this size may vary depending on the conditions, which may influence the compression of the data, the size of each run may be larger or smaller than the average. An impression of this variability is given in the last column which shows the standard deviation normalised by the average. These results clearly show that by far the largest contribution is from the G2G forecast runs, despite the modest number of runs in the database. The G2G historical runs are much smaller – but given that there are more runs in the database, the total size is the largest. It can also be seen that the Delft3D runs are sizable, in particular the forecast run, where a run is also made every six hours.

Table 17 shows the same statistics for the OC database. It can be seen that there is much more variability. Inspection of individual runs shows that there is much less data for some of the runs, while others are comparable in size to those on the FSS. It is as yet unclear what leads to these differences. What is clear is that the same runs dominate on the OC as on the

FSS. In both databases, the large sizes are due to the gridded data and much less due to the scalar data,

Table 18 finally shows the data contributed by the import workflows. Again it is clear that the gridded data dominates, despite a short expiry time. Table 19 shows how the different module instances in the import gridded data workflow contribute to the size of the database in the OC and the FSS. Note the large differences for some of these module instances. There are again large differences, likely due to synchronisation profiles.

These statistics have been established using queries 5 and 6.

Table 16 Overview of important forecast workflows in The FEWS system (FSS Database)

Workflow	**Size in database (Mbytes)**	**Number of runs (-)**	**Average size per run (KBytes)**	**Normaliised stdev (%)**
Coastal_Forecast	2.80	40	72	2.27
Fluvial_Forecast	9.78	40	286	11.1
Fluvial_Historical	1.78	10	212	10.0
Delft3D_Forecast	20.3	40	518	9.4
Delft3D_Historical	1.85	10	189	3.0
National_Forecast	786	4	201267	22.5
National_Forecast_MOGREPS	469	2	240100	2
National_Historical	2254	20	123469	12
PerformanceMonitoring	0.69	10	60.8	20

Table 17 Overview of important forecast workflows in The FEWS system (OC Database)

Workflow	**Size in database (Mbytes)**	**Number of runs (-)**	**Average size per run (KBytes)**	**Normaliised stdev (%)**
Coastal_Forecast	0.49	7	72	1.6
Fluvial_Forecast	1.7	36	18.7	323
Fluvial_Historical	0.84	9	96	113
Delft3D_Forecast	3.55	9	101	199
Delft3D_Historical	1.85	9	89	113
National_Forecast	322	4	82474	100
National_Forecast_MOGREPS	508	4	260319	0.4
National_Historical	490	5	101504	16
PerformanceMonitoring	0.69	10	58.0	21

Table 18 Overview of data contributed by import workflows

Workflow	**Size in database (Mbytes)**	**Number of runs (per day)**	**Average size per day (MBytes)**
ImportExternal	126	144	1.656
ImportExternalGrids	217	144	110

Table 19 Overview of data contributed by different module instances in import grids workflows

Module Instance	**FSS Size in database (Mbytes)**	**OC Size in database (Mbytes)**
CatchmentAveraged_ObservedRainfall	1330	2
ImportNimrod	281	45
ImportMOTemperature	100	96
ImportNWPExt	95	53
ImportUKPP	32	2
ImportMOGREPS	24	19
CatchmentAveraged_ForecastRainfall	18	0

3.3.9 Volume of warm states in the FSS database

Warm states are stored as binary objects in the warm states table of the database. Where models have large states, such as gridded models, then storing these states may also take substantial space. Table 20 provides an overview of the size of the states in the FSS database. An asterisk is used to indicate this is for all module instances of that type. Again it is clear that the G2G model dominate. The total size of the WarmStates table can be seen to be quite substantial.

Table 20 Overview of the size of states from different models that store states

Module Instance	**Number of states in database (-)**	**Total Size in database (Kbytes)**	**Average size per state**
ARMA*	40	7	188
Delft3D_Historical	10	1410	144338
Grid_National_Historical	20	25526	1306921
ISIS*	670	910	1391
KW*	4	21	5314
PDM*	740	151	209

3.3.10 Volume of module datasets in the FSS database

The final table that is investigated is the module Instance datasets. This table contains zipped representations of the models for distribution. Where multiple versions of a large dataset are kept then this may impact database size.

4 Discussion and Recommendations

4.1 General comments the database size

The assessment presented in the previous chapter provides some clear indicators that can be used to propose reductions of the database size. This section will provide recommendations that can be expected to lead to significant reductions in database sizing.

Recommendations will be made for individual module instances. In this section the prime contributors to the database size will be considered. Additional to recommendation on the size, these will also be made to improve synchronisation speed, in particular where large amounts of data may be synchronised to remote offices with low bandwidths.

The analysis shows that the time series table is by far the largest contributor to the database size. Though modest in contribution, recommendation to keep the size of other tables to a limit will also be made. Finally possible enhancements to the FEWS software will be made that will help reduce the size of the database and/or improve the performance of synchronisation.

4.2 Recommendations on external historical time series

This section discusses sources that contribute to the size of the external historical time series in the database. General recommendations are made for the sources that have the largest contribution.

Grids with Observed rainfall data

Analysis: The grids from some sources dominate the size of the FSS database. In some cases these grids are not available for viewing (only scalars of the same module instances are displayed). The gridded data is currently kept in the database for 15 days (default expiry time). This data is sampled from the radar data, and subsequently processed in the same module instance to form multiple scalar time series available for viewing.

Recommendation (P1):

- Create a new module instance that only re-samples the gridded data to a smaller grid
- Include this module new module instance both in the Import workflow, as in the forecast and historical workflow.
- Set the output gridded data to temporary. Given the size of the grid also set the expiry time in the time series set to the order of 1 hour to avoid data that is flushed to the database due to memory limitations residing in the database for too long.

Functional Impact:

- This change is not expected to have any functional impact, while reducing the database size by some 33-35%. The workflows will need to run an additional module instance but the performance impact is expected to be marginal.

Gridded rainfall data converted into different units

Analysis: Due to the general adapter not being able to do some conversions on import of data, an intermediate step is introduced in both the Forecast and Historical Grid model workflows. The data is ultimately combined through a switch into a merged precipitation product that is then used in the model. It is also available for viewing through the spatial display. The added value of this data being visible is considered limited, as the raw data is already available. It would seem to constitute an intermediate step and the raw data is available for viewing. The data is currently only synchronised using a specific synchronisation profile.

Recommendation (P1):

- Remove this data from viewing in the spatial display.
- If required the values can be changed for viewing by adding a divider to the time series set.
- Set the output gridded data to temporary. Given the size of the grid also set the expiry time in the time series set to the order of 1 hour to avoid data that is flushed to the database due to memory limitations residing in the database for too long.

Functional Impact:

- This change is not expected to have any functional impact except that the data itself is not visible. However, the raw data is visible. If required the values can be changed for viewing by adding a divider to the time series set.
- This reduction will have significant impact on the OC database and synchronisation speeds.

Interpolated input data for the Grid model

Analysis: Intermediate time series used in the grid model forecast and historical runs should not be stored. It is not available for viewing, and the module instance that creates it is run in all National forecasts (Forecast, Historical and MOGREPS).

Recommendation: (P1)

- Set the output gridded data to temporary. Given the size of the grid also set the expiry time in the time series set to the order of 1 hour to avoid data that is flushed to the database due to memory limitations residing in the database for too long.

Functional Impact:

- This change is not expected to have any functional impact.

Imported Temperature grids

Analysis: There are four parameters imported from a specific data source, though only one of these is external historical. The grid has a large domain, with temperature only over the land surface being relevant. While visualising the data provides nice information it does come at the cost of a large amount of data. These data are used in the grid model run, and the expiry time is currently set to 2 days. The total database size is 7 MB which is not considered excessive, but the same issue as for the external historical grids holds for the external forecasting grids.

Recommendation (P1):

- Set up a mask that results in the only the temperature values over the land-surface being saved with all other values set to missing. While this need not change the domain, missing values are compressed with a factor of 99%. This can be done through a module instance with the PCRaster transformation, but this is complex and has some disadvantages. A minor adaptation in the software that applies a mask on import of the data is recommended.

Functional Impact:

- No impact on the running of models if the mask is correctly set up. The only impact is on a smaller domain of the temperature data being visible.

4.3 Recommendations on external forecasting time series

This section discusses module instances that contribute to the size of the external forecasting time series in the database. Recommendations are only made for those activities that have the largest contribution and are generalized.

Imported Forecast rainfall grids

Analysis: There are three parameters imported from this source, all of which are forecast data. The grid has a large domain and a high resolution (2km). These data are used in the Grid model run, and the expiry time is currently set to 2 days. The total database size is quite large. The data is available in the spatial display, and although the domain is larger than the land area where the data is used by the model, it is informative for the forecaster to see the rainfall pattern. The data comes from three source models, the 1.5km grid model, the 12 km grid model and the 25 km grid model. These are imported at 15 min, 1 hour and 3 hour resolutions. The database has a similar ratio of about 30:2:1. All data are imported at the same grid resolution, though the source models have a much smaller size. One option to reduce the size is to import these data in their original resolution. However, the ratio of sizes already shows that this will not contribute too much as the largest contribution will remain. An easy reduction of size would not seem easy to achieve.

Recommendation:

- No specific recommendations

Functional Impact:

- None.

Expected development:

- Increasing the resolution from the current 2 km would result in increases in the size occupied by this time series. NWP resolutions are often increased following model updates - so it is important to keep this is mind. Doubling the resolution from the current would quadruple the space taken by this dataset which may make it an issue.

Imported Surge Grids

Analysis: Raw surge data is used by several models and is therefore important. The resolution of the grid is quite low, but the domain is large, which though informative to some extent does come at the expense of taking quite some space. Reducing the domain could easily reduce the space occupied. Note that there are already no values over the land area.

Recommendation (P1):

- Apply a mask to save values over a reduced domain - or alternatively transfer a smaller domain from the source.

Functional Impact:

- No impact on the running of models if the mask is correctly set up. The only impact is on a smaller domain of the temperature data being visible.

Imported Rainfall now cast grids

Analysis: The raw input data from the STEPS now cast is used by the Grid model forecast. Though this data is more or less the same resolution as the other rainfall grids, a new forecast is available every 15 minutes. This means that the volume of data is high. The length of the forecast on the other hand is low (6 hours). The expiry time is now set to 2 hours, but as the FSS only runs a rolling barrel task on a 6-hourly basis the effective time spent in the database is longer. Increasing the frequency of FSS rolling barrel runs will decrease the size. This is also clear on the OC - where the size of this dataset is small by comparison.

Recommendation (P2):
- Increase the frequency of rolling barrel runs on the FSS servers to reduce the size of the stored data.

Functional Impact:
- None

Imported rainfall ensemble grids

Analysis: The raw ensemble data is of low resolution both spatially and temporally. However, the ensemble size of 24 contributes to the size of the data in the data stored. The domain is again quite large, covering the whole of the UK, and a large part of the North Sea. This domain could be easily reduced by half without loosing much of the information content.

Recommendation (P2):
- Reduce the size of the domain for which the MOGREPS data is stored.

Functional Impact:
- None

Imported wind and wave grids

Analysis: The raw Wind and Wave input data is used in various tidal and coastal forecasts. There are four parameters imported, two for wind (u and v components) and two for wave (height and direction). Though the grid resolution is relatively low, the domain of the forecasts is large, extending across the full North Sea and a good part of the Atlantic.

Recommendation (P1):
- Reduce the size of the domain for which the data is stored, preferably using a mask on import.

Functional Impact:
- No impact on the running of models if the mask is correctly set up. The only impact is on a smaller domain being visible.

Expected development:
- At the time of writing an increase in the resolution and lead time of these data was being contemplated. This could increase the size of the dataset. To limit this increase it is recommended to consider saving values only over the necessary domain.

4.4 Recommendations on simulated historical time series

All major contributions to the simulated historical time series in the database are from the grid model historical state runs. This is some 99% of the total database size of the database for this time series type. The following recommendations are generalized.

Interpolated composite of rainfall data grids

Analysis: Processed data used in the grid model run as an input to the run, but also in viewing. The raw data sources of which the data is a composite of are also available for viewing individually. It would seem logical to remove the data from viewing. A simple enhancement to the spatial display can help viewing as a composite with no loss of functionality but a very significant database size reduction.

Recommendation (P1):

- Remove the data from the spatial display, set the time series set to temporary in the process workflow, as well as short expiry time. Applying the mask to reduce the size of the raw data stored will also be of benefit here.

Functional Impact:

- None (provided functionality in Spatial Display extended and reconfigured). Otherwise small as raw data components available.

Processed data for grid model run

Analysis: Processed data used in the grid model run as an input to the run. It is not available for viewing. Some of the data in the module instance that creates the data is set to temporary, but for some time series it is saved. It would seem that all could be set to temporary.

Recommendation (P1):

- Set all data from this module instance to temporary. Also set a short expiry time to avoid records flushed from memory persisting too long.

Functional Impact:

- None.

Output from Grid model run

Analysis: The stored results of the grid model runs for the historical period includes several parameters, of which discharge is stored at 15 minute resolution and other data at 3 hourly resolutions. Interestingly most data cannot be viewed in the spatial display. The table below shows that the discharge data dominates, though in absolute terms the other data is also substantial.

Table 21 Size of G2G model run results for the historical period (FSS Database).

Parameter	**Size ofdataObject (MBytes)**	**Percentage of total (%)**
Q.simulated.historical	277.1	86%

Soilmoisture.simulated.historical	16.0	5%
Soilsaturation.simulated.fraction	12.7	4%
Snow.simulated.wet / Snow.simulated.dry	16.1	5%

Recommendation (P1):

- Remove the discharge Q.simulated.historical from the database by setting time series to temporary (this data is post processed for viewing later on in the workflow)
- Remove the two snow parameters from the database by setting time series to temporary (this data is post processed for viewing later on in the workflow)
- Consider re-sampling the soil moisture information to (sub) catchments and storing related scalar time series. It would seem the information content of this is quite low, except to give a regional impression.
- Reduce the expiry time of all data from the grid model workflows to e.g. 10 days.

Functional Impact:

- Small.

Post-processed results from Grid model run

Analysis: Post-processed results from the grid model runs for the historical period are saved in the database. Some 80% of the data is the post-processed discharge data, while 20% is taken in by the warning levels. Both are the results shown in the spatial display. As these data are the main outputs of the G2G runs a reduction would seem difficult.

Recommendation (P3):

- Consider saving the data at hourly intervals rather than 15 minute intervals, though this will result in a loss of resolution.
- Shorten expiry time of historical grid model runs to 10 days.

Impact

- Some loss of information, but limited in forecast process.

Pre-processed data as input for grid model run

Analysis: Data is displayed as a composite, as well as the raw inputs and can therefore be easily set to temporary. If not then it should be set to simulated forecasting.

Recommendation (P1):

- Remove from database by setting to temporary (or set to simulated forecasting if it is to be retained)

Impact

- None (provided functionality in Spatial Display extended and reconfigured). Otherwise small as raw data components available.

Recalculated snow data from grid model run

Analysis: This is primarily the recalculated snow data, and contains a prime output of the grid model run. The only recommendation to reduce without loss of too much information is to shorten expiry times.

Recommendation (P2):
- Reduce expiry times of the grid model run.

Impact
- Small.

4.5 Recommendations on simulated forecasting time series

All major contributions to the simulated forecasting time series in the database are from the gridded model forecast runs. The following recommendations are generalized.

Gridded output from ensemble run

Analysis: Gridded outputs of the ensemble runs are stored in the database. However, the data cannot be seen the spatial display, as only post-processed results are shown.

Recommendation (P1):
- Set to temporary and remove from database

Impact
- None.

Post-processed results from Grid model run

Analysis: The post processed results of the grid forecasts contain the discharge grid and the warning level grid. Again some 80% is the discharge data and the remaining 20% the warning levels. The expiry time of these forecasts is currently set at 2 days which seems reasonably short.

Recommendation (P3):
- Consider saving the data at hourly intervals rather than 15 minute intervals, though this will result in a loss of resolution.

Impact
- Loss of resolution.

4.6 General Configuration recommendations

In analysing the configuration it was found that in some case the expiry times of gridded data differed. This was primarily for external forecast grids. These expiry times could be made as uniform as possible, and also compared against the expiry times of the forecast model runs that use the data. Also synchLevels should be made uniform

4.7 Recommendations for additional developments in Delft FEWS

In the recommendations made for reducing the data of the different time series several enhancements to the software have been identified. These recommendations are provided below:

4.7.1 Resolve issues of MC-MC synchronisation

It was found that import data was saved twice – with the obvious impact on database sizes. Data imported from external sources independently by each master controller should not be synchronised between databases. This concerned synchLevel 23, but may also concern other synchLevels and should be carefully analysed and corrected where wrong. The MC-MC synchronisation configuration will need to be adjusted to resolve these issues.

4.7.2 Resolve issues with unit conversion on import of data using the general adapter

Additional data to be saved and complexity in configuring modules seems to be introduced by the General Adapter not supporting unit conversions. A simple extension of the general adapter should resolve this.

4.7.3 Implement the ability to save grids using a mask

Several of the grids considered used a grid domain quite a bit larger than the area of interest. By saving values only for a mask area the size of the values saved can be reduced significantly. An example could be to save temperature values only over the land surface areas as it is only there that they are used. The mask could have a smaller domain to reduce the domain and missing values for grid cells not needing to be saved.

4.7.4 Improve spatial display to show composites of different (raw) data sources

To avoid needing to save merged time series of raw inputs, extend the spatial display to allow combining time series which are then merged on the fly. Only the raw data then needs to be saved and as these are usually external time series types can be stored with much less space.

4.7.5 Create an interface to allow for more flexible synchronisation of time series

With the large datasets from e.g. gridded models, the synchronisation between different parts of the system is very complex. The synchronisation is generally quite complex and mistakes are easily made. This may mean that some users without sufficient rights to see gridded forecasts do find themselves synchronising this data – resulting in large synchronisation times and large database volumes.

To make this process more transparent, a more interactive and configurable approach to synchronisation can be easily developed. It is proposed to develop a two layer approach, where synchronisation profiles are no longer set up in the root configuration, but moved to the regional configuration. A display (perhaps an extension of the current system monitor) can then be used to make visible how the channels are configured, and if permissions allow, change synchronisation settings).

4.8 Summary of recommendations

Implementing the recommendations above can be expected to reduce database sizes by The FEWS system to 70-80% of the current size.

The recommendations can be divided into different priorities. In some cases very significant savings can be achieved without any impact on functionality.

5 Conclusions

This report provides an analysis of the sizing of the FEWS system databases. Both on the Forecasting Shell Server and on the Operator Clients, the size of these have become excessive as the complexity of the forecast models and the resolution and sources of data imported increased.

The analysis has identified the data that contributes most to these excessive sizes and recommendations made on how the database volumes can be reduced.

In some cases the reduction can be achieved through simple configuration changes. In others small updates of the Delft FEWS software can be implemented to help achieve significant reductions in database size. Many of these changes will have no or little impact on the functionality provided, while enhancing forecast system performance.

If these, or in any case the most important recommendations are implemented it is expected that the size of the operational databases will be drastically reduced. Though an exact calculation has not been made, it is expected that the sizes can be reduced by up to 70% of the current size.

Appendix

Appendix 1: SQL Queries used

Query 1: Counts the number and size of records according to time series type

```
SELECT Count(blobid) as NumBlobs, TimeSeriesType, ValueType, Sum(blobsize) as
BlobSize FROM TimeSeries GROUP BY TimeSeriesType;
```

Query 2: Counts the number and size of records according to value type (scalar, long., or grid)

```
SELECT Count(blobid) as NumBlobs, ValueType, Sum(blobsize) as BlobSize FROM
TimeSeries GROUP BY ValueType;
```

Query 3: Counts the number and size of records of a selected time series type. Substitute the ID in the query for different time series types, and orders by the module instance that these are attributed to.

External Historical	:	0
External Forecasting	:	1
Simulated Historical	:	2
Simulated Forecasting	:	3
Temporary	:	4

```
SELECT ModuleInstanceId, ValueType,  Count(blobid) as NumbBlobs, Sum(blobsize)
as BlobSize FROM TimeSeries WHERE TimeSeriesType=ID GROUP BY ModuleInstanceID,
ValueType;
```

Query 4: Counts the number of records in the time series table that have a blob (time series) smaller than a specified size.

```
SELECT ModuleInstanceId, ValueType,  Count(blobid) as NumbBlobs, Sum(blobsize)
as BlobSize FROM TimeSeries WHERE BlobSize < 64 GROUP BY ModuleInstanceID,
ValueType;
```

Query 5: Overview of tasks in the task table, and related task-id. Note that tasks with an ID without an underscore are normally scheduled tasks submitted from the Administrator Interface

```
SELECT WorkflowID, TaskID, Count(TASKID) as NumbTasks FROM TASKS Group by
WorkflowId, TaskID;
```

Query 5: Calculate size of time series data added to database by all runs in the database of a specific workflow ID, sorted by moduleInstanceId. Substitute workflow_id with the relevant workflow id (name).

```
SELECT ModuleInstanceId,Count(blobid) as NumBlobs, Sum(blobsize) as BlobSize
FROM TimeSeries WHERE creatortaskRunId IN (SELECT tr.TaskRunId FROM TaskRuns tr,
tasks t WHERE t.workflowid Like '%<workflow_id>%' AND t.taskid = tr.taskid)
GROUP BY ModuleInstanceId
```

Query 6: Count the number of runs and the data contributed by each run for selected workflow id.

```
SELECT creatortaskRunId,Count(blobid) as NumBlobs, Sum(blobsize) as BlobSize
FROM TimeSeries WHERE creatortaskRunId IN (SELECT tr.TaskRunId FROM TaskRuns tr,
tasks t WHERE t.workflowid Like '%<workflow_id>%' AND t.taskid = tr.taskid)
GROUP BY creatortaskRunId
```