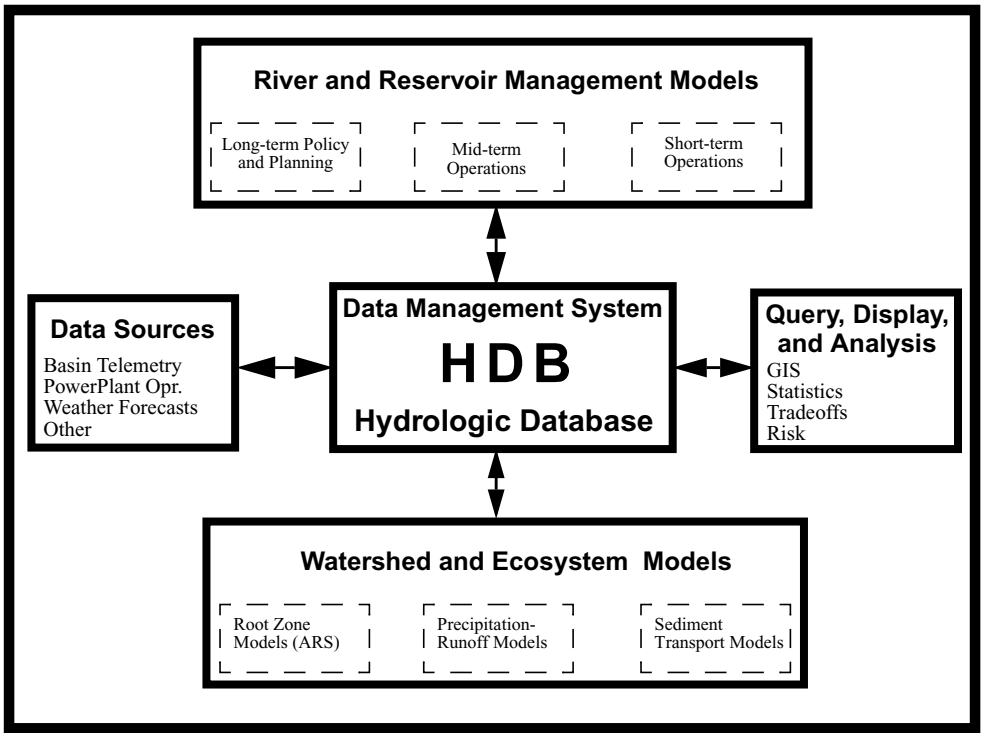
# Hydrologic Database (HDB): Derivation Application User Guide

Next Page




The Center for Advanced Decision Support for Water and Environmental Systems (CADSWES)
University of Colorado at Boulder
College of Engineering and Applied Science
Department of Civil, Environmental and Architectural Engineering
Campus Box 421
Boulder, CO 80309-0421
Phone: (303) 492-3972 Fax: (303) 492-1347
E-mail: inquiries@cadswes.colorado.edu
http://cadswes.colorado.edu

CADSWES

# Section 1 Intended Audience

This document describes the functionality of the Derivation Application and is intended for all users who are implementing the Derivation Application in their HDB databases. Users should have a detailed knowledge of their base data and how it needs to be manipulated after reaching the database.

Next Page

Previous Page

# Section 2 Application Overview

The Derivation Application enables the automatic derivation of data values provided to HDB by external sources. Its purpose is to take data that has come into HDB and apply some method to it (like average or maximum value) to derive the data up to longer time intervals. Based on specifications defined by the user for each site datatype id, the Derivation Application automatically populates data into all tables representing longer time intervals, as applicable, including hour, day, month, year, and water year. In this way, all data in the time series tables of HDB are kept up to date and internally consistent, resulting in an accurate data repository across all time intervals .

Next Page

Previous Page

# Section 3 Permissions

As discussed in the next section, the Derivation Application is run with the user named *app_user*. When running, the Derivation Application invokes a special role named *derivation_role*. This role controls permissions in the database and allows the Derivation Application to interact with the Time Series Data Tables to delete, update, and insert derived values.

Next Page

Previous Page

# Section 4 Invoking the Application

The Derivation Application is a single executable file named *derivation*, located in the HDB bin directory with other HDB application executables. It is invoked as a batch process on the Unix command line along with user name and password. The user name is *app_user* and the password is the name of the database that you are connecting to. A sample invocation is as follows:

Next Page

derivation app_user <dbName>

The application can be manually invoked through the command line, although more likely it will be configured to run automatically as part of a *cron* job on the Unix system to derive data on a regular schedule.

Previous Page

When invoked, the application creates a temporary table named *r_interval_update* in the *app_user*'s schema; the application uses this table in processing the derivations and then deletes it at the end of the run. Multiple versions of the Derivation Application cannot run correctly on a database at the same time. If the *r_interval_update* table exists, the derivation application is prevented from starting a new run, effectively creating a check so that multiple versions of the application cannot run simultaneously. In the event the Derivation Application crashes during execution, the *r_interval_update* table will need to be deleted manually before the application can be invoked again. The user receives the following informational message if the existence of the *r_interval_update* table is preventing the application from running:

> Derivation application is either already running on this database or crashed and did not properly drop the *r_interval_update* table. If the program crashed, the *r_interval_update* table under the *app_user*'s schema must be dropped before the derivation application will rerun.

For the application to operate correctly, a number of environment variables must be set. These are included in the file named *.cshrc_hdb_app,* which sets up the general environment for HDB.

# Section 5 Inputs

## Command Line

Application name, user name, and password are input on the command line. (See Section 4 “Invoking the Application” on page 5.)

Next Page

Previous Page

## Files

None

## Dialog Boxes

None

## Database Driver Tables

Two sets of tables drive the Derivation Application. One set contains derivation specifications that indicate how calculations should be performed for each site datatype and its derivations. Users enter these specifications through the Meta Data Application. The forms-based interfaces for entering and maintaining Derivation Application specifications are discussed in detail in the Meta Data Application User Guide.

The other set of tables involves the loading of data into HDB. When data are loaded into the HDB base area table *r_base*, an update table called *r_base_update* is automatically populated by a database “trigger” (a pre-programmed database function). The Derivation Application uses *r_base_update* to identify what new data need to be derived since the previous run of the application. The update table requires no user input, as the program automatically maintains the update table.

# Section 6 Outputs

## Data Files

None.

Next Page

## Log and Error Files

Previous Page

The Derivation Application uses a file named *derivation.log* to record messages and errors. The file is appended to if it already exists and is created if it does not. By default it is placed in the *$HDB_ENV/log/* directory. The user can specify a different directory by assigning an environment variable named *DERIVATION_LOG_PATH* to the desired path.

When a Derivation Application run starts, it writes a message to the log file to that effect with the start time. During execution, the application will write informational messages to the log file concerning any non-normal data conditions. If an error condition is encountered that terminates the application run, the error is recorded in the log file. If the application finishes successfully, a message is posted indicating that the run with the specified start time has completed along with the elapsed time and the number of records processed.

## Database

The Derivation Application's primary outputs are to the Time Series Data Tables in HDB. These currently include the following tables:

- *r_instant*
- *r_other*
- *r_hour*
- *r_day*
- *r_month*
- *r_year*
- *r_wy*

The *r_other* table is an addition to this group and is meant to hold data at a regular interval not represented in one of the other tables (i.e., quarterly) or data that comes in at irregular intervals. As the application copies and derives new data that has entered the database, it inserts, updates, and deletes records from the Time Series Data Tables to maintain a complete and consistent data record of time series values.

# Section 7 Using Application Functionality

## Introduction

The Derivation Application fulfills the requirement of fully populating the time series tables with consistent data values. Users set up specifically how data are derived to populate these tables. The following sections describe the derivation process and the various types of functionality available to users.

Next Page

Previous Page

# Base Area

Prior to the Derivation Application, data were loaded directly into the applicable time series tables in HDB by various loading applications, creating an inconsistency problem between tables. A key to using the Derivation Application is having a base area from which all time series data comes into HDB. All data is inserted, updated, or deleted from the table *r_base*, and the Derivation Application takes care of propagating the data through the time series tables according to specifications defined by the user. The columns in *r_base* and their descriptions are as follows:

Next Page

Previous Page

- *site_datatype_id* - the site datatype id for the data record.
- *interval* - the time interval of the data record.
- *start_date_time* - start time of the data record.
- *end_date_time* - end time of the interval the data represents. Same as start time for instantaneous data.
- *value* - value of the data record.
- *agen_id* - id for the organization responsible for the data record. Not propagated into the time series tables by the Derivation Application.
- *overwrite_flag* - indicates if the record is an overwrite value.
- *date_time_loaded* - time the record was loaded into *r_base*. Populated automatically by a trigger.
- *validation* - any validation flag assigned to the record by the source system. Not propagated into the time series tables by the Derivation Application.
- *collection_system_id* - id for the system that collected the data. Not propagated into the time series tables by the Derivation Application.
- *loading_application_id* - id for the application that loaded the record into *r_base*. Not propagated into the time series tables by the Derivation Application.
- *method_id* - id to identify any particular method that was applied outside of HDB to generate the data record (i.e., data point average). Not propagated into the time series tables by the Derivation Application.
- *computation_id* - id to identify any particular computation that was applied outside of HDB to generate the data record. Not propagated into the time series tables by the Derivation Application.

The first four fields, *site_datatype_id*, *interval*, *start_date_time*, and *end_date_time* form the primary key for the table. It is up to the application loading data into *r_base* to supply the desired information to populate the applicable columns for the data record. Incoming data records that have incomplete foreign keys are automatically rejected by HDB.

A table *r_base_update* implements the mechanism for tracking what new or modified data needs to be propagated by the Derivation Application. Subject to the table's constraints, any data can be inserted into *r_base*. In order to be processed by the Derivation Application, the inserted or modified record has to get recorded into *r_base_update*. A trigger does this automatically—but only if the record is marked

as an overwrite record or if the site datatype id of the record has a derivation specification written for it. Data that do not meet the criteria of being overwrites or which have a specification simply remain in the base area without any propagation to the Time Series Tables.

The *r_base* table has a *date_time_loaded* field that is automatically populated with a trigger when new or updated data come into the table. When a data record is modified or deleted from r_*base*, a copy of the earlier record is saved to the table *r_base_archive* along with the date of the change. This dating and archiving process permits the recreation of all data values in *r_base* at any past time.

Next Page

Previous Page

# Derivation Specifications

Derivation specifications allow users to define how their base data loaded into HDB get propagated into the Time Series Tables. Users enter derivation specifications through the Meta Data Application. A section in the Meta Data Application User Guide titled “Maintaining Derivation Application Driver Data” gives detailed instructions about entering specifications into the forms interface. That information will not be repeated in this document; rather, discussion in subsequent sections will focus on describing the functionality represented by the specifications.

Next Page

Derivation specifications are maintained in two driver tables in the database. The *ref_derivation_source* table contains information relating to source data from *r_base* used in derivations. A site datatype can have only a single entry in *ref_derivation_source*. This entry specifies a single time interval for data in *r_base* for the site datatype. This is the interval (probably the smallest interval) at which data regularly arrives for this site datatype in *r_base*. The entry in *ref_derivation_source* causes base values with this interval to be written to *r_base_update* and copied to the appropriate time series table by the Derivation Application. For example, if data for a site datatype are specified as coming in at an hourly interval, only hourly data with this site datatype are processed as base data and copied to the *r_hour* table by the Derivation Application. Data for any other time interval for this site datatype are not processed unless they are marked as overwrite values. If a site datatype does not have an entry in *ref_derivation_source*, data for it remain in *r_base*, but never get propagated into the Time Series Tables unless they are marked as overwrite values.

Previous Page

The *ref_derivation_destination* table contains the specifications for deriving values upward through the Time Series Tables after base values are copied from *r_base* due to a *ref_derivation_source* specification. There can be multiple destination entries for a single source site datatype. For example, destination specifications for average and maximum values can be created for a set of base data.

Any change in a derivation specification, either source or destination, will cause the old specification to be archived with the date and time of the change. The two relevant archive tables are named *ref_derivation_source_archive* and *ref_derivation_destination_archive*. The Derivation Application records the date and time that any value is derived into the Time Series Tables. This date, along with current and archived specifications, permits the determination of the exact specification under which the value was derived.

# Overwrite Values

Overwrite values are values that get moved into the Time Series Tables by the Derivation Application without the existence of a derivation source specification. They could be values for a site datatype that has no source specification, or values for a site datatype in *ref_derivation_source*, but at a different interval than the one specified for base area data. This functionality is provided to easily allow "better" data to be overwritten into a time series or to allow data not normally processed to be copied into the Time Series Tables without having to write a specification.

Next Page

Overwrite values are identified in the base area by an *O* flag in the *overwrite_flag* field of the data record. The Derivation Application automatically copies the record to the corresponding slot in the Time Series Table indicated by the interval field of the record. Since there is no specification, there is no checking of the magnitude of the data value as is possible with a *ref_derivation_source* specification. If users create an overwrite record, they are responsible for ensuring that the value is valid.

Previous Page

The Derivation Application marks the record copied to the Time Series Table with an *O* in the *overwrite_flag* field of the table. The overwrite value creates a discontinuity with data in the next shorter interval since it was not derived from those data. Any data in the shorter interval that would have derived to the overwrite value's interval are marked with a *T* in the *overwrite_flag* field to signify that data continuity is truncated at this interval. In this way data on both sides of the discontinuity are identified and marked.

The Derivation Application is designed so that an overwrite value will not get overwritten if new data comes in at a shorter interval and new values are derived upward. For example, provisional hourly data can enter the database and be derived up through all of the time intervals to water year. Final daily values can later be overwritten into the day interval, and all intervals longer than day will be recalculated based on the final daily values. If a change is made to one of the provisional hourly values, it is recognized that data continuity is truncated at the hour interval, so the daily overwrite value will not get replaced by recalculated provisional hourly data.

# Derivation Functionality

If derivation specifications exist, inserts, updates, and deletes of data in the base area results in calculation or recalculation of dependent data in the Time Series Tables. In this way, all time series data are kept internally consistent and up-to-date with the source data.

Derivation is a batch process, and each run of the Derivation Application is assigned a single date time that is used to time stamp the *date_time_loaded* field of all time series data derived during that run. This permits an easy determination of when values were derived and which ones were derived together.

Next Page

The existence of a *ref_derivation_source* specification for a site datatype triggers copying of the data from the base area to the Time Series Table representing the interval of the data. To legitimately represent a valid value for a noninstantaneous interval, the *start_date_time* of the base data must be greater than or equal to the start time of the slot in the interval table, and the *end_date_time* of the base data must be less than or equal to the end time of the slot in the interval table. For example, an hour value in *r_base* with start and end times of 9:20 to 10:20 would not get copied out of *r_base* because it does not legitimately represent either the 9:00 to 10:00 or the 10:00 to 11:00 hour slots in the *r_hour* table. If there are two legitimate values in *r_base* for an interval that is not instantaneous, the one with the latest time stamp in the *date_time_loaded* column is copied by the Derivation Application. Restated, the most recently loaded value for an interval is assumed to be the best one. Screening of data during copying is discussed below. (See “Maximum and Minimum Value Specifications” on page 14.).

Previous Page

After the copy step, *ref_derivation_destination* specifications cause the derivation of values upward through the Time Series Tables according to specified methods. For regular business intervals (hour and above), the first derivation is always to the next longer interval. For base data with an interval of instant or other, the user specifies the first interval that data is derived to. For example, very sparse instantaneous data could be derived directly to monthly values as the first step. All derived data from a source is derived to the same first destination interval. That is, you may derive maximum, minimum, and average data from instantaneous data, but if the first destination is set to day, all of these data will be created initially for the day interval.

From the first destination interval, data is automatically derived upward through all the longer time series tables to year and water year. This guarantees that all the tables are fully populated to the extent possible. Each subsequent interval is derived from the data in the next shorter interval. For example, monthly values are calculated from the derived daily values rather than from the original hourly values that came into the database as the base interval data.

The following subsections describe the functionality of specifications that can affect the derivation process.

## Time Offset in Minutes

Users are allowed to specify a time offset in minutes that applies to all data of a site datatype that enters the base area at its normal interval. This is available to handle

delays in reporting from telemetry devices where the delay is actually reflected in the time stamp of the data. For example, a 10:00 reading from an instrument may normally get reported and time stamped as 10:10 due to delays in the telemetry system. The time offset allows for a correction to be specified and applied to determine the actual time of the reading. Providing an offset does not redefine the length of an interval, but merely shifts the interval's time for this site datatype.

For regular business intervals, hour and longer, the time offset gets applied in screening data for copy from the base area to the correct slot in the associated interval table. An offset of 10 minutes for hourly data means that legitimate data in the base area for the 9:00 to 10:00 hour slot in the *r_hour* table must have a start time of greater than or equal to 9:10 and an end time less than or equal to 10:10.

Next Page

For data of intervals instant and other, the offset gets applied when deriving to the first destination interval. For example, an offset of 10 minutes for this case means that instantaneous data must fall between 10 minutes after the beginning of the hour and 10 minutes after the end of the hour to be used in deriving an average value for the hour. In this example, if maximum and minimum values were also specified to be derived for this site datatype, they would use the same 10 minute offset.

Previous Page

## Maximum and Minimum Value Specifications

Maximum and minimum value specifications are used as filters to screen out bad data values. These optional validation parameters are supplied by the user for a particular site datatype and are applied as screening criteria when that data is copied out of *r_base*. The four parameters are maximum and minimum expected value and maximum and minimum cutoff value. The behavior is that if a value in *r_base* is less than the minimum cutoff value or greater than the maximum cutoff value, the value is not copied out of *r_base* to its associated interval table and is not used in any subsequent derivations. In this event, a message is written to the log file noting the specifics of this record, the criteria it did not meet, and the fact that it was not copied out of *r_base*. This message can alert the user to potential sensor or telemetry malfunctions in the data collection system.

If a value is less than the minimum expected value, but not less than the minimum cutoff value, it is copied out of the base area to the corresponding interval table and is marked with an *l* for low in the *derivation_flags* field of that table. Similarly, if a value is higher than the maximum expected value, but not higher than the maximum cutoff value, it is copied to its interval table and assigned an *h* for high in the *derivation_flags* field.

All screened values after the copy step, including those marked as *low* or *high*, can be used as source data for subsequent derivations. The *low* and *high* flags are not propagated to subsequently derived values.

Note: If any, or all, of the minimum and maximum value specifications are not supplied by the user, the Derivation Application simply does not perform the corresponding check.

## Time Window Specifications

Time window specifications supplied by the user are applied in deriving beginning

of period and end of period values for a site datatype from a shorter interval to a longer interval. The purpose of the window is to restrict where in the interval a source value must fall to be considered valid as a beginning or end of interval value. Windows can be specified for each applicable interval as values for a site datatype are derived upward through the Time Series Tables. The windows for an interval are defined in units of time corresponding to the next shorter interval—e.g., windows for derivations to hour are specified in minutes, and windows for derivations to month are specified in days.

Windows are specified separately for *beginning of period* and for *end of period* derivations for a site datatype. A specification for one of these derivations includes a required window and a desired window. An example to demonstrate the functionality is that an instantaneous value should ideally fall within the last 5 minutes of an hour (desired window) but must fall within the last 15 minutes of an hour (required window) to be propagated as the end of hour reading. If the propagated value fell outside of the desired window but within the required window, it is marked with a *w* in the *derivation_flags* field to show it was outside the desired window. The *w* flag is not propagated to data in any subsequently derived intervals.

Next Page

Previous Page

Note: If two or more legitimate data values are found in an *end of period* window, the one with the latest date time is propagated as the *end of period* value. Similarly, the earliest value of multiple legitimate values is the one propagated as the *beginning of period* value.

The windows specified for end of period derivations are calculated to be inclusive back from the end of the interval. For beginning of period derivations, the window times are inclusive forward from the beginning of the interval. The window times cannot be greater in length than the interval the user is deriving to (i.e., windows for derivations to hour cannot be greater than 60 minutes).

Window times are referenced from the interval boundaries as defined or redefined. In the case of a 10-minute source data time offset for this site datatype (See "Time Offset in Minutes" on page 13.), an *end of period* window of 15 minutes before the end of the hour would be interpreted as from 5 minutes before the clock hour to10 minutes past the clock hour, since the offset hour actually ends 10 minutes past the clock hour. Intervals can also be redefined at the installation level. (See "Interval Redefinitions" on page 21.) If the day is redefined as 8am to 8am instead of as midnight to midnight, a *beginning of period* window of one hour on the day interval would be interpreted as from 8am to 9am.

Note: Desired and required window specifications can be supplied, or not, for any interval. If a window specification is not supplied by the user, the Derivation Application does not perform the corresponding window time check.

## Number of Source Value Specifications

If the derivation method is not *beginning of period* or *end of period* where time window specifications are used (See "Time Window Specifications" on page 14.), then the user can specify the number of source values needed to create data for this site datatype at a longer interval. This specification gives the user control over the conditions under which enough data is present to legitimately derive a value to the next interval. Minimum desired and minimum required number of source data values can be specified for each applicable interval as values for a site datatype are

derived upward through the Time Series Tables.

For example, the user may desire that 20 (minimum desired) hourly values be present to derive a daily value but will allow a daily value to be derived if as few as 16 (minimum required) values are present. If the number of hourly values is less than 20 but greater than or equal to16, the derived daily value will be marked with an *n* in the *derivation_flags* field to indicate that the desired number of source values was not present.

Since months have an unequal number of days as source data points, users can specify the required and desired number of data points for derivations to month as a number of missing values. This is indicated by making the specification negative. For example, specifying the required number of source values as -2 means that to derive a month value you need 29 or more day values for January, 28 or more day values for June, and 26 or more day values for February in a non-leap year.

Next Page

Previous Page

If an *n* flag is assigned to a derived value, the flag does not get propagated to any subsequently derived interval that uses this value as source data. Required and desired number of source value specifications are optional for all intervals. If a specification is not supplied by the user, the Derivation Application does not perform that check on the number of source values.

## Partial Calculations

Since it may be desirable for the user to see derived data for intervals that are not yet complete, the user can specify that partial calculations be performed by the Derivation Application for a particular derivation of values up through the time intervals. For example, under a partial calculation an *average day* value will be calculated even though it is noon, an *average month* value will be calculated even though it is only the 15th, and *average year* and *average water year* values will be derived even though it is only July. When a value is derived part way through an interval, it is marked with a *p* in the *derivation_flags* field to indicate that this value is from a partial calculation. When the interval is complete, the partial value is either made final (based on all the data that came in) or is removed (if the appropriate data is not available under the specifications to generate a final value).

If a partial calculation is not specified by the user, no value is derived for an interval until the interval is complete.

## Derivation Flags

The functionality that assigns flags to the *derivation_flags* field of records in the Time Series Tables has been discussed above under the relevant sections. Because these flags are assigned within the Derivation Application code, the user cannot modify or make additions to the flags. More than one flag, if applicable, can be assigned to a data record. For easy reference, a summary of the flags and their

meanings is presented below.

**TABLE 1. Derivation Flags**

| Flag Code | Flag Name | Explanation |
|---|---|---|
| h | high | Value was higher than expected. |
| l | low | Value was lower than expected. |
| n | number | Value had fewer than the desired number of source values. |
| p | partial | Value was the result of a partial calculation. |
| w | window | Value was outside of the desired time window. |
| M | manual | Value was not derived by the Derivation Application, but results from some other process operating in the Time Series Tables. |

Next Page

Previous Page

# Derivation Methods

A *ref_derivation_source* specification for a site datatype causes values to be copied out of *r_base*. Then *ref_derivation_destination* specifications determine how data gets derived upward through the Time Series Tables from the copied base values. Each destination specification indicates the exact calculation method to be used for its derivation. A method must be specified to derive a value to a longer time interval because somehow a number of values at the shorter interval must be combined to a single value at the longer interval (e.g., by taking the average of the values).

Next Page

When the Derivation Application places any data into the Time Series Tables, it marks the *method_id* field of the data record with the id of the method responsible for generating the value. This is the id for the copy method in the case of data being copied from *r_base*. Otherwise, it is the id of the exact calculation method specified in the destination specification responsible for generating the value.

Previous Page

The following table summarizes the methods currently available in the Derivation Application. The term "source interval" in the method descriptions refers to the interval of time where source data points for the derivation can be found. For example, when deriving instantaneous values to hourly data with a ten-minute offset applied to the site datatype (See "Time Offset in Minutes" on page 13.), the source interval for deriving a 9:00 to 10:00 value would be 9:10 to 10:10. All instantaneous data in this source interval would be source data values for the derivation.

**TABLE 2. Derivation Methods**

| Method Id | Method Name | Method Description |
|---|---|---|
| 2 | Data-point average of values | Sum of the data points in the source interval divided by the number of data points. |
| 3 | Time-weighted average using linear interpolation | Uses the latest data point in the previous interval and the earliest value in the source interval to linearly interpolate a beginning of source interval value. Uses the earliest data point in the following interval and the latest data point in the source interval to linearly interpolate an end of source interval value. The two interpolated end points and the data points in the source interval are connected. The area under the resulting line is divided by the length of the interval to generate a time-weighted average. If no data is available in adjacent intervals, no value is calculated. |
| 4 | Data-point beginning of period | The earliest data point in the source interval. |
| 6 | Copy of data | Data was copied from *r_base*. |
| 7 | Data-point end of period | The latest data point in the source interval. |

**TABLE 2. Derivation Methods**

| Method Id | Method Name | Method Description |
|---|---|---|
| 8 | Linearly-interpolated end of period | Finds the latest data point in the source interval and the earliest data point in the following interval and linearly interpolates a value for the end time of the source interval. If no data is found in the following interval, no result is calculated. |
| 9 | Data-point flow to volume | For derivation to the first destination interval, finds the average flow of all data points in the source interval and multiplies by the length of the interval to get a volume. For subsequent derivations, it sums the volume data points in the shorter source interval to generate a volume for the longer interval. |
| 11 | Data-point maximum | The largest data point value in the source interval. |
| 12 | Data-point minimum | The smallest data point value in the source interval. |
| 17 | Data-point sum over time | Sum of the data points in the source interval. |
| 19 | Volume to Volume | Sums the volume data points in the source interval to generate a volume for the destination interval. |

Next Page

Previous Page

# Compounding of Methods

The Derivation Application supports single compounding of methods, meaning that two derivation methods are applied to the source data. For example, instantaneous temperatures could be derived with the data-point maximum method to create maximum temperatures for intervals up through the Time Series Tables. The daily maximum temperatures could then be derived with the data-point average method to create average daily maximum temperatures for month and longer time intervals. The interval is important in the specification of the source data for the compound derivation. Average daily maximum temperatures for the month represent completely different data than average hourly maximum temperatures for the month, so it must be specified that maximum temperatures at the day interval are the source data for the compound derivation.

Next Page

Previous Page

Multiple compounding—where more than two methods are applied to source data—is not supported in the Derivation Application. Conceptually, an example of this would be average for the month of the maximum hourly end of period value for each day.

# Interval Redefinitions

An interval redefinition means that the start and end times of the interval are shifted from the default start and end times. This functionality in HDB lets users redefine any time interval on an installation-wide basis. For example, the default times for the day interval of midnight to midnight could be redefined for a particular HDB installation to be 8:00am to 8:00am. If an interval is redefined in this way, all greater time intervals take this into account. In this example, the month would run from 8:00am on the first day of one month to 8:00am on the first day of the next month.

Next Page

The Meta Data Application provides a form for entering interval redefinition specifications. The interval to be redefined is selected (hour, day, month, year, or water year), a time offset is entered (number can be positive or negative), and the legitimate offset unit is then provided as a selection (unit for the next smaller time interval). The offset can be thought of as . . . "By how much and in what direction do I want to move the start time of the interval from its normal default start time?"

Previous Page

Water year is treated as a redefined year and the specification for this comes already populated in HDB. Water year has a time offset of -3 with offset units of months. This means that the start time for a water year is moved back three months from the normal start of a year—from January 1 to October 1. Therefore, water year runs from October 1 to October 1 of the next year.

In the example of the redefined day mentioned above, the specification would be for day, the offset would be +8, and the offset unit would be hours. The direction one moves the offset can be important. A redefinition of day by -16 hours would also make the day run from 8:00am to 8:00am; however, in this case the month would run from 8:00am on the day before the first day of the month and run through 8:00am on the last day of the month.

# Section 8 Examples

This section presents an example of a derivation so that the user can see the implementation of some of the functionality described in earlier sections. Sample *ref_derivation_source* and *ref_derivation_destination* specifications are shown. A small amount of base data is presented along with how this data is propagated through the Time Series Tables under the given specifications.

Next Page

## Sample *ref_derivation_source* specification

Previous Page

The following *ref_derivation_source* specification is for a *site_datatype_id* of 20323, which represents instantaneous flow data. Data of this type will be copied from *r_base* to the *r_instant* table subject to the minimum and maximum value parameters shown. Any subsequent derivation of these data will initially be to the hour interval. No time offset is specified for this *site_datatype_id*.

**TABLE 3. Sample *ref_derivation_source* specification**

| Table Columns | Data |
|---|---|
| **SITE_DATATYPE_ID** | 20323 |
| **INTERVAL** | instant |
| **FIRST_DESTINATION_INTERVAL** | hour |
| **MIN_VALUE_EXPECTED** | 0 |
| **MIN_VALUE_CUTOFF** | 0 |
| **MAX_VALUE_EXPECTED** | 100 |
| **MAX_VALUE_CUTOFF** | 150 |
| **TIME_OFFSET_MINUTES** | |

## Sample *ref_derivation_destination* specifications

The following table shows two *ref_derivation_destination* specifications that are associated with the site datatype id of 20323. These specifications indicate how data of this type will be derived upward through the Time Series Tables from the *r_instant* table.

Specification 1 derives data from site datatype id 20323 in *r_instant* to site datatype id 20329 in *r_hour* and other longer Time Series Tables. This destination datatype represents maximum flow and the *method_id* of 11 indicates that the data-point maximum method will be used to derive these data. Partial calculations will not be performed for this derivation. Many of the other potential columns for specifying derivation parameters do not apply to this case (i.e., compounding and windows), so are blank. Number of source value parameters can apply and are specified here for derivations to hour. It is desired to have four instantaneous values for derivation to

hour, but there must be at least three in order to derive an hourly value. The number of value parameters for intervals longer than *hour* could be filled out; however, they are left blank in this specification. This means that no checking will be performed on *number* of source values for derivations to these intervals.

Specification 2 derives data from site datatype id 20323 in *r_instant* to site datatype id 20325 in *r_hour* and other longer Time Series Tables. This destination datatype represents average flow and the *method_id* of 2 indicates that the data-point average of values method will be used to derive these data. Partial calculations will be performed for this derivation. No hourly or longer time interval parameters are entered for number of source values.

Next Page

Previous Page

**TABLE 4. Sample *ref_derivation_destination* specifications**

| Table Columns | Spec 1 | Spec 2 |
|---|---|---|
| **BASE_SITE_DATATYPE_ID** | 20323 | 20323 |
| **DEST_SITE_DATATYPE_ID** | 20329 | 20325 |
| **METHOD_ID** | 11 | 2 |
| **PARTIAL_CALC** | | Y |
| **COMPOUNDING_SOURCE_SDI** | | |
| **COMPOUNDING_SOURCE_INTERVAL** | | |
| **HR_DESIRED_EOP_WINDOW** | | |
| **HR_REQUIRED_EOP_WINDOW** | | |
| **HR_DESIRED_BOP_WINDOW** | | |
| **HR_REQUIRED_BOP_WINDOW** | | |
| **HR_DESIRED_NUMBER_SOURCE** | 4 | |
| **HR_REQUIRED_NUMBER_SOURCE** | 3 | |
| **HR_WINDOW_UNIT** | | |
| **DAY_DESIRED_EOP_WINDOW** | | |
| **DAY_REQUIRED_EOP_WINDOW** | | |
| **DAY_DESIRED_BOP_WINDOW** | | |
| **DAY_REQUIRED_BOP_WINDOW** | | |
| **DAY_DESIRED_NUMBER_SOURCE** | | |
| **DAY_REQUIRED_NUMBER_SOURCE** | | |
| **DAY_WINDOW_UNIT** | | |
| **MON_DESIRED_EOP_WINDOW** | | |
| **MON_REQUIRED_EOP_WINDOW** | | |
| **MON_DESIRED_BOP_WINDOW** | | |
| **MON_REQUIRED_BOP_WINDOW** | | |
| **MON_DESIRED_NUMBER_SOURCE** | | |
| **MON_REQUIRED_NUMBER_SOURCE** | | |
| **MON_WINDOW_UNIT** | | |

TABLE 4. Sample *ref_derivation_destination* specifications

| Table Columns | Spec 1 | Spec 2 |
|---|---|---|
| YR_DESIRED_EOP_WINDOW | | |
| YR_REQUIRED_EOP_WINDOW | | |
| YR_DESIRED_BOP_WINDOW | | |
| YR_REQUIRED_BOP_WINDOW | | |
| YR_DESIRED_NUMBER_SOURCE | | |
| YR_REQUIRED_NUMBER_SOURCE | | |
| YR_WINDOW_UNIT | | |
| WY_DESIRED_EOP_WINDOW | | |
| WY_REQUIRED_EOP_WINDOW | | |
| WY_DESIRED_BOP_WINDOW | | |
| WY_REQUIRED_BOP_WINDOW | | |
| WY_DESIRED_NUMBER_SOURCE | | |
| WY_REQUIRED_NUMBER_SOURCE | | |
| WY_WINDOW_UNIT | | |

Next Page

Previous Page

## Sample *r_base* data

The following table presents some sample *r_base* data for site datatype id 20323. (Note: Not all of the columns in *r_base* are shown here. Only those that are key to this derivation example are presented.) Instantaneous data coming in every 15 minutes are shown for three hours on January 1st. In addition, there is one *day* value for site datatype id 20329 for January 2nd that is marked with on overwrite flag.

TABLE 5. Sample *r_base* data

| SITE_DATA TYPE_ID | INTERVAL | START_DATE_ TIME | END_DATE_ TIME | VALUE | OVERWRITE _FLAG |
|---|---|---|---|---|---|
| 20323 | instant | 01-JAN-2003 01:00 | 01-JAN-2003 01:00 | 60 | |
| 20323 | instant | 01-JAN-2003 01:15 | 01-JAN-2003 01:15 | 80 | |
| 20323 | instant | 01-JAN-2003 01:30 | 01-JAN-2003 01:30 | 100 | |
| 20323 | instant | 01-JAN-2003 01:45 | 01-JAN-2003 01:45 | 120 | |
| 20323 | instant | 01-JAN-2003 02:00 | 01-JAN-2003 02:00 | 80 | |
| 20323 | instant | 01-JAN-2003 02:15 | 01-JAN-2003 02:15 | 60 | |
| 20323 | instant | 01-JAN-2003 02:30 | 01-JAN-2003 02:30 | 300 | |
| 20323 | instant | 01-JAN-2003 02:45 | 01-JAN-2003 02:45 | 300 | |
| 20323 | instant | 01-JAN-2003 03:00 | 01-JAN-2003 03:00 | 80 | |
| 20323 | instant | 01-JAN-2003 03:15 | 01-JAN-2003 03:15 | 60 | |
| 20323 | instant | 01-JAN-2003 03:30 | 01-JAN-2003 03:30 | 40 | |

**TABLE 5. Sample *r_base* data**

| SITE_DATA TYPE_ID | INTERVAL | START_DATE_ TIME | END_DATE_ TIME | VALUE | OVERWRITE _FLAG |
|---|---|---|---|---|---|
| 20323 | instant | 01-JAN-2003 03:45 | 01-JAN-2003 03:45 | 300 | |
| 20329 | day | 02-JAN-2003 00:00 | 03-JAN-2003 00:00 | 95 | O |

## Sample *r_instant* data

Next Page

Previous Page

The following table shows *r_instant* data for site datatype id 20323 that was copied over from *r_base* as a result of the *ref_derivation_source* specification. Not all columns of *r_instant* are shown—only those relevant to this example. All values have a *method_id* of 6 to indicate a copy of data. The one row with a value of 120 was given a derivation flag of *h*, meaning that it was higher than the maximum value expected of 100, but was still less than the maximum cutoff value of 150. Three rows from *r_base* that had values of 300 were not copied to *r_instant* because they exceeded the maximum cutoff value of 150.

**TABLE 6. Sample *r_instant* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20323 | 01-JAN-2003 01:00 | 01-JAN-2003 01:00 | 60 | 6 | | |
| 20323 | 01-JAN-2003 01:15 | 01-JAN-2003 01:15 | 80 | 6 | | |
| 20323 | 01-JAN-2003 01:30 | 01-JAN-2003 01:30 | 100 | 6 | | |
| 20323 | 01-JAN-2003 01:45 | 01-JAN-2003 01:45 | 120 | 6 | h | |
| 20323 | 01-JAN-2003 02:00 | 01-JAN-2003 02:00 | 80 | 6 | | |
| 20323 | 01-JAN-2003 02:15 | 01-JAN-2003 02:15 | 60 | 6 | | |
| 20323 | 01-JAN-2003 03:00 | 01-JAN-2003 03:00 | 80 | 6 | | |
| 20323 | 01-JAN-2003 03:15 | 01-JAN-2003 03:15 | 60 | 6 | | |
| 20323 | 01-JAN-2003 03:30 | 01-JAN-2003 03:30 | 40 | 6 | | |

## Sample *r_hour* data

The following table shows data that would be derived from *r_instant* to the *r_hour* table under the sample *ref_derivation_destination* specifications. Not all of the columns of *r_hour* are shown.

Specification 1 above generates the 20329 site datatype id values with method 11, data-point maximum. No hourly value is derived for the 2:00 to 3:00 hour because only two source values are available, which does not meet the required number of source values of 3 for derivations to hour. The 3:00 to 4:00 hour has a derivation flag of *n*, meaning that the desired number of source values of 4 were not available,

but the required number of 3 were.

Specification 2 above generates the 20325 site datatype id values with method 2, data-point average of values. Hourly values are derived for all three hours because there are no parameters specified for desired and required number of source values for derivations to hour in this specification.

**TABLE 7. Sample *r_hour* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20329 | 01-JAN-2003 01:00 | 01-JAN-2003 02:00 | 120 | 11 | | |
| 20329 | 01-JAN-2003 03:00 | 01-JAN-2003 04:00 | 80 | 11 | n | |
| 20325 | 01-JAN-2003 01:00 | 01-JAN-2003 02:00 | 90 | 2 | | |
| 20325 | 01-JAN-2003 02:00 | 01-JAN-2003 03:00 | 70 | 2 | | |
| 20325 | 01-JAN-2003 03:00 | 01-JAN-2003 04:00 | 60 | 2 | | |

Next Page

Previous Page

## Sample *r_day* data

The following table shows data derived from *r_hour* assuming that the current time is some time later in January. Not all of the *r_day* columns are shown.

Specification 1 creates the January 1st value for site datatype id 20329. It is the maximum of the two hour values that are available for that day. There are no desired or required number of source values specified for derivations to day to limit this derivation. The January 2nd value is created by the overwrite entry for this day in *r_base*. This value gets copied directly to the day table, so receives the *method_id* of 6 for copy of data. This entry gets flagged with an *O* in the *overwrite_flag* column to indicate it is an overwrite value.

Specification 2 creates the January 1st value for site datatype id 20325. It is the average of the three hour values available for this day. There are no desired or required number of source values specified for derivations to day to limit this derivation.

**TABLE 8. Sample *r_day* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20329 | 01-JAN-2003 00:00 | 02-JAN-2003 00:00 | 120 | 11 | | |
| 20329 | 02-JAN-2003 00:00 | 03-JAN-2003 00:00 | 95 | 6 | | O |
| 20325 | 01-JAN-2003 00:00 | 02-JAN-2003 00:00 | 73 | 2 | | |

### *Sample r_month* data

Assuming that it is sometime later is January, only a value for site datatype id 20325 is derived for the month. This is because Specification 2 allows for partial calculations to be performed before the interval is over. This partial month calculation is flagged with a *p* in the *derivation_flags* field to indicate that this value is partial. Not all of the columns in the *r_month* table are shown.

Next Page

Previous Page

**TABLE 9. Sample *r_month* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20325 | 01-JAN-2003 00:00 | 01-FEB-2003 00:00 | 73 | 2 | p | |

### Sample *r_year* data

Similar to the partial month value, a partial year value is created for site datatype id 20325. This shows the average flow so far this year even though it is only some time in January. Not all of the columns in the *r_year* table are shown.

**TABLE 10. Sample *r_year* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20325 | 01-JAN-2003 00:00 | 01-JAN-2004 00:00 | 73 | 2 | p | |

### Sample *r_wy* data

Assuming that no month values for site datatype id 20325 are available for October, November, and December of 2002, the following partial calculation value would be created to show the average flow so far for the water year. If monthly values were available for the other months they would also be averaged into the partial calculation of average so far in the water year. Not all of the columns in *r_wy* are shown.

**TABLE 11. Sample *r_wy* data**

| SITE_DATA TYPE_ID | START_DATE_ TIME | END_DATE _TIME | VALUE | METH-OD_ID | DERIVA-TION_ FLAGS | OVER-WRITE_ FLAG |
|---|---|---|---|---|---|---|
| 20325 | 01-OCT-2002 00:00 | 01-OCT-2003 00:00 | 73 | 2 | p | |

Next Page

Previous Page

# Section 9 FAQ (Frequently Asked Questions)

This section addresses frequently asked questions about the derivation application.

**What is this error message about *r_interval_update* that I get when trying to start the Derivation Application?**

The existence of the *r_interval_update* table in the *app_user*'s schema acts as a check to insure that only one copy of the Derivation Application is running on a database. Getting this message means that the application is either already running or has crashed during mid-run and needs to have the *r_interval_update* table deleted manually. (See "Invoking the Application" on page 5.)

Next Page

Previous Page

**Why isn't my data from *r_base* going into the Time Series Tables?**

There are several possible answers to this question:

1. For data coming into *r_base* for a site datatype at its normal interval, a *ref_derivation_source* specification must be entered into the database for data to be copied to the appropriate Time Series Table. (See "Derivation Specifications" on page 11.)
2. If data comes in for a site datatype at an interval other than the normal one specified in the *ref_derivation_source* specification, it will not be copied to the Time Series Tables unless it is marked as an overwrite value. (See "Overwrite Values" on page 12.)
3. If data has a *ref_derivation_source* specification and comes in at the normal interval named in the specification, it may be getting screened out based on the magnitude of the values. In a *ref_derivation_source* specification, the user can optionally specify minimum and maximum cutoff values so that data outside of the cutoff range is not copied out of *r_base*. (See "Maximum and Minimum Value Specifications" on page 14.) If this does happen, a message is written to the log file showing the record that did not get copied and the reason why. (See "Log and Error Files" on page 7.)
4. If data has a *ref_derivation_source* specification and comes in at the normal non-instantaneous interval named in the specification, it would not be copied out of *r_base* if its *start_date_time* and *end_date_time* spanned across two intervals as represented in the Time Series Table to which it was to be copied. (See "Derivation Functionality" on page 13.)

**Why are the values in the water year table dated as January 1 instead of October 1?**

For *water year* to be adjusted to the correct start and end date of October 1, an entry in the *ref_interval_redefinition* table must be present. (See "Interval Redefinitions" on page 21.) This entry is normally created by the scripts that create HDB; however, if it gets removed, water year records will be *calculated and dated* incorrectly. In this case, the entry must be recreated as described in the Interval Redefinition discussions.