

E-1. adbinit suite

1.0 General Information

1.1 Application Description

The adbinit suite consists of the following scripts and applications:

```
run_adbinit (Korn script)
ingestdef (ecpg/C)
locatdef (ecpg/C)
load_ingest (Korn script)
load_locat (Korn script)
```

This suite is used to initially define the location and ingestfilter tables of the archive database, and may be used to as a way to insert new definitions into one or both tables. The run_adbinit script executes the two ecpg/C programs, ingestdef and locatdef. These two applications generate load files for the RFC Archive DB. The ingestdef application reads the IHFS database's ingestfilter table and the locatdef application reads the IHFS database's location table. Once the load files have been reviewed, the user then runs the load_ingest and load_locat scripts; these scripts utilize the pgload command to load the data into the RFC Archive database's location and ingestfilter tables. All applications take advantage of the apps_defaults system.

1.2 Enhancements/Bug Fixes/Changes

Build OB7.2

The application was redesigned to work with Postgres RDBMS. In addition, it is assumed that an RFC may be using this application suite to update existing location and ingestfilter tables, not for initializing these two tables for the very first time.

Build OB6

For this build a work around had been created since the RFC Archive DB is in Informix RDBMS and the IHFS DB is in Postgres RDBMS. The script and applications that unload data from the IHFS DB (which, beginning with ob6, is in Postgres) must be installed on a box other than the RAX. The run_adbinit script copies the files to the appropriate place on the RAX. The user can then logon to the RAX and complete the process. In addition, the access to the histdata db in the locatdef program was eliminated. The locatdef and ingestdef were converted to ecpg/C from esql/C at this time.

Build OB5

Bug Fixes

Modified ingestdef application and dbload command file to take into account build OB4 adb *ingestfilter* table change. Column “p” was dropped.

1.3 Design Considerations

An integral difference between the location tables of the IHFS database and archive database is that multiple entries for the same location identifier (lid) are allowed for in the archive database so that a history of meta data on a location can be accumulated.

1.4 Assumptions application makes

ingestdef

To aid the reader who may be unfamiliar with the archive database table definitions, Attachment A contains a cross reference between the IHFS database and archive database *ingestfilter* tables.

For the archive database’s *ingestfilter* table, the following fields are defined as null: “det”, “obstime”, “ownag” and “ownloc”.

locatdef

To aid the reader who may be unfamiliar with the archive database table definitions, Attachment B contains a cross reference between the IHFS database and archive database *location* tables.

For the archive database’s *location* table, the following fields are defined as null: “goes”, “huc”, “zon” and “dbsource”.

For the archive database’s “name” and “det” fields, the standard is for the values to be in mixed case. The program assumes the IHFS database values for “name” and “det” are in upper case, thus the program converts the IHFS “name” and “det” fields to mixed case. For the archive database’s “name” field, the “name” and “detail” fields of the IHFS database are concatenated.

The “sbd” is set as follows: 1) to the “sbd” value of the IHFS database, if one exists, 2) if “sbd” does not exist, then it is set to the value of the “lrevise” field in the IHFS database, and 3) if neither the “sbd” or “lrevise” exists, it defaults to “today’s date”.

If inadequate data is available to generate a “sbd” value, the “sbd” defaults to “07/04/1776”.

The “sed” is set to null.

For no post sites, the “sbd” is defined in similar manner to a post site and the “sed” is always set to null.

In the archive database the longitude is stored with a negative sign.

In the archive database the elevation is stored as an integer instead of a float. If no elevation is available the value defaults to zero.

For sites where the state code in the IHFS database is “XX”, the application looks at the lid to see if the code can be determined by the last 2 characters of the NWSLI lid. If no state is still defined then the application looks at the first two digits of the COOP station number if one exists to determine the state code.

If the county name is unavailable in the IHFS database, then the countyfips code field defaults to “XXX” for the archive database.

For sites where the post flag is set to “1” in the IHFS database *location* table, the post field in the archive database is set to “2”.

It is assumed that the IHFS database’s *rfc* table contains the 5 character acronyms for the offices; for example ABRFC, APRFC. For the archive database the first two letters of this entry are used in the “rfc” field of the archive database’s *location* table.

For the “country” field, the value is initialized to “US”, but the code then uses the state field to determine if any locations are in Canadian provinces or Mexican states based on the 2-char abbreviations the author has run by the field (see table 1). If it is determined that a location is in a Canadian province, the country field is set to “CA”; for Mexico the country field is set to “MX”.

2.0 Configuration Information

2.1 Apps_defaults tokens

Four apps_defaults tokens are used by the various programs and scripts, these are:

db_name	name of the IHFS database on dx1
pghost	name of the system IHFS database is on
adb_name	name of the archive database on the archive system
rax_pghost	name of the system archive database is on

The tokens *db_name* and *pghost* are used by both the ingestdef and locatdef programs. The token *adb_name* and *rax_pghost* are used by the run_load-data script.

3.0 User How-To

Initializing the *location* and *ingestfilter* tables of the archive database is as easy as one, two, three, etc. The steps are as follows:

- ☐ login in as oper on the archive system
- ☐ cd to /rfc_arc/bin/adbinit
- ☐ execute the run_adbinit script (this will run both the ingestdef and locatdef programs)
- ☐ review the output files ingestfilter.unl and location.unl (optional)
- ☐ if the location and ingestfilter tables have previously been defined and this suite is being used to “update” the tables, the shef_decode_raw application should be stopped before proceeding (optional)
- ☐ execute the load_ingest script (this will load new entries into the *ingestfilter* table)
- ☐ execute the load_locat (this will load new entries into the *location* table)
- ☐ review the error output files. (For the *ingestfilter* table it is likely that the error file will have many entries indicating duplicate rows, this is normal and is not a problem. The error file for the *location* table should be zero in size.) (optional)
- ☐ you are now ready to start posting data to the RFC Archive database via its shefdecoder.

4.0 Troubleshooting Information

Log files are generated by the locatdef and ingestdef applications when run by the run_adbinit script. Check these log files for problems. The important part of this log file information is generally the lines related to database name and sql code values when the programs are not finding data. Excerpts from successful run with the debug option turned on are shown in sections 4.1 and 4.2 for each program.

If further help is needed, contact the RFC Support Group.

4.1 Sample output for ingestdef

```
hd5_12krf@ONLINE
sqlca.sqlstate open database 00000
select
lid,pe,dur,ts,extremum,ts_rank,ingest,ofs_input,stg2_input from
ingestfilter order by 1;
sqlca.sqlstate prepare 00000
sqlca.sqlstate declare 00000
sqlca.sqlstate open cursor 00000
sqlca.sqlstate fetch cursor 02000
sqlca.sqlstate close cursor 00000
```

```
sqlca.sqlstate free cursor 00000
sqlca.sqlstate close IHFS database 00000
```

```
ingestdef finished
```

4.2 Sample output for locatdef

```
today is 06/05/2006 6 5 6
hd_ob6ounx
sqlca.sqlstate open database 00000
select
lid,name,detail,det,lat,lon,elev,location.state,has.,location.wf
o,rfc,sbd,countynum,post,lrevise from location,counties where
location.county=counties.county and location.state=couties.state
sqlca.sqlstate prepare 00000
sqlca.sqlstate declare 00000
sqlca.sqlstate open cursor 00000
sqlca.sqlstate fetch cursor 02000
sqlca.sqlstate close cursor 00000
sqlca.sqlstate free cursor 00000
sqlca.sqlstate close IHFS database 00000

locatdef finished
```

Attachment A

archive database ingestfilter table		IHFS database ingestfilter table		Description
column	datatype	column	datatype	
lid	varchar(8)	lid	varchar(8)	location identifier
pe1 pe2	char(1) char(1)	pe	varchar(2)	SHEF Physical Element codes
dur	char(1)			SHEF Duration code
idur	smallint	dur	smallint	numeric value of SHEF Duration code
t s	char(1) char(1)	ts	varchar(2)	SHEF Type-Source codes
e	char(1)	extremum	varchar(1)	SHEF Extremum code
ts_rank	smallint	ts_rank	smallint	numerical ranking of alternate SHEF TS codes for the same location and parameter.
det	varchar(40)			descriptive detail
ingest	char(1)	ingest	varchar(1)	post data to database, archive db 0 - no & 1 - yes ihfs_db F - no & T - yes
new_report	char(1)			new entry? Y or N
active	char(1)			active sensor? Y or N
ofs_input	char(1)	ofs_input	varchar(1)	feed data to OFS, archive db 0 - no & 1 - yes ihfs_db F - no & T - yes
obstime	time w/o time zone			nominal obs time for monthly data
ownag	varchar(6)			owner agency
ownloc	char(3)			owner agency location
mpe_input	char(1)	stg2_input	varchar(1)	feed data to stagell/AreaWide archive db 0 - no & 1 - yes ihfs_db F - no & T - yes
primary key lid, pe1, pe2, dur, idur, t, s, e		primary key lid, pe, dur, ts, extremum		

Attachment B

archive database location table		IHFS database location table		Description
column	datatype	column	datatype	
lid	varchar(8)	lid	varchar(8)	location identifier
sbd	date	sbd or lrevise	date	begin date (mm/dd/ccyy)
sed	date			end date (mm/dd/ccyy)
goes	varchar(8)			dcp platform id
name	varchar(60)	name	varchar(50)	station name archive db - city name with distance and direction ihfs_db - name is city name with no state, distance or direction
det	varchar(40)	det	varchar(30)	additional station description information
lat	float	lat	float	latitude
lon	float	lon	float	longitude
elev	integer	elev	float	elevation in ft msl
state	char(2)	state	varchar(2)	2-char PO state code
huc	varchar(8)	hu	varchar(8)	hydrologic unit code
countyfips	char(3)			county fips code
zon	char(4)			NWS zone code
has	char(3)	has	varchar(3)	hydrologic service area WFO identifier
wfo	char(3)	wfo	varchar(3)	WFO identifier
post	smallint	post	integer	post flag
bsource	char(3)			
rfc	char(2)	rfc	varchar(5)	RFC acronym (example: MB instead of MBRFC)
countryfips	char(2)			country fips code (example: US for United States)
primary key lid, sbd		primary key lid		