Observational DataBase (ODB*) and its usage at ECMWF

anne.fouilloux@ecmwf.int

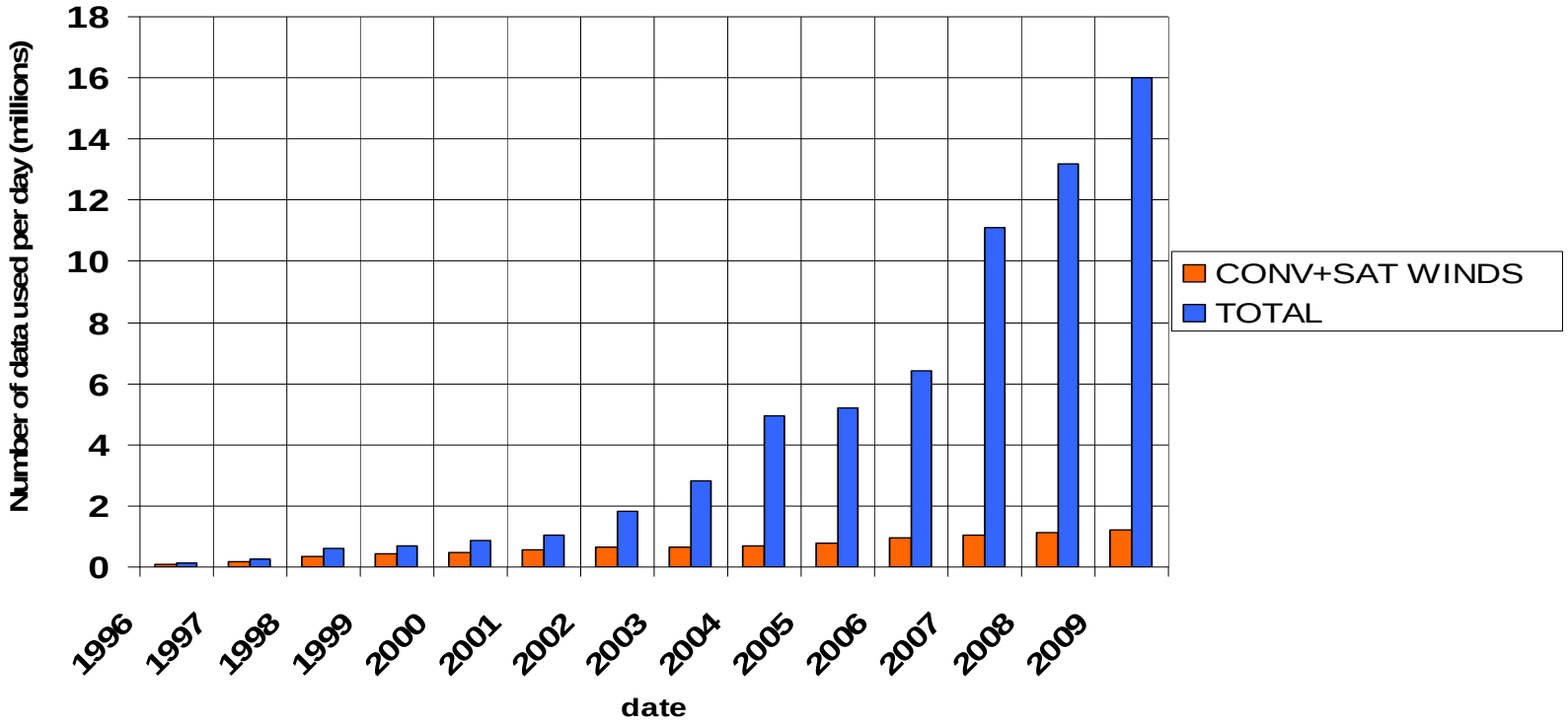
**ODB has been developed and maintained by Sami Saarinen*

Outline

- **Observational usage over the past decades at ECMWF**
- **Before ODB…**
- **Observational DataBase (ODB)**
  - **What is ODB?**
  - **And what is NOT ODB!**
- **its current usage in IFS**
- **The way forward**

Observational usage over the past decades

- **One of the major progress made over the last two decades in numerical weather prediction (NWP) can be attributed to the improved utilization of observations.**

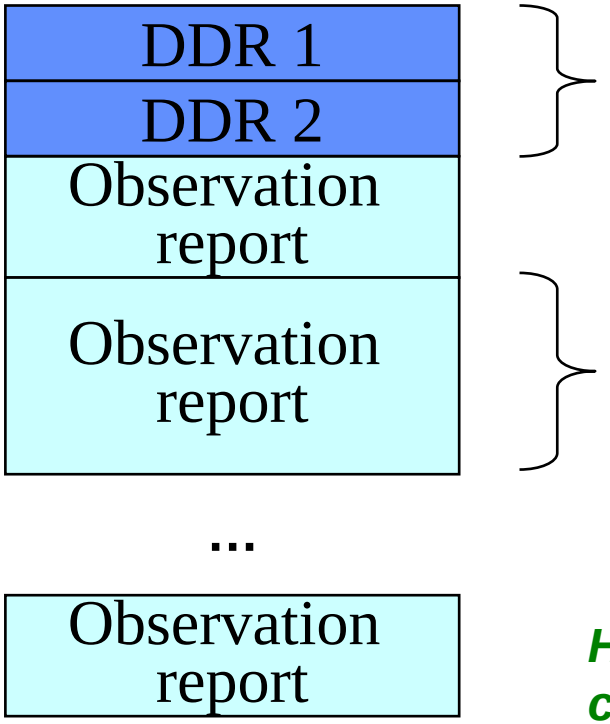


- **But this has been possible only thanks to the usage of supercomputers as well as the development of efficient strategies to read/write/process these observations.**

First step toward an efficient strategy ...

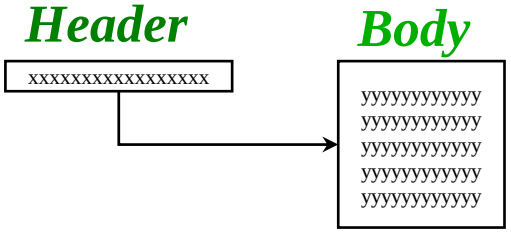
CMA (Central Memory Array) file structure

- **Based on encoding all data into IEEE 64 bit floating points.**
- **Once read, CMA were kept in memory for a fast data access.**



Data Description Record (fixed length)

Each observation report (variable length) consisted of two parts:



Header: identification, position and time coordinates, etc.
Body: observation value, pressure levels, channel numbers, etc.

With the introduction of 4D var in IFS
and the growing number of satellite observations ...

→ There was a need for a new approach to store and access observational data

ODB (Observational DataBase)

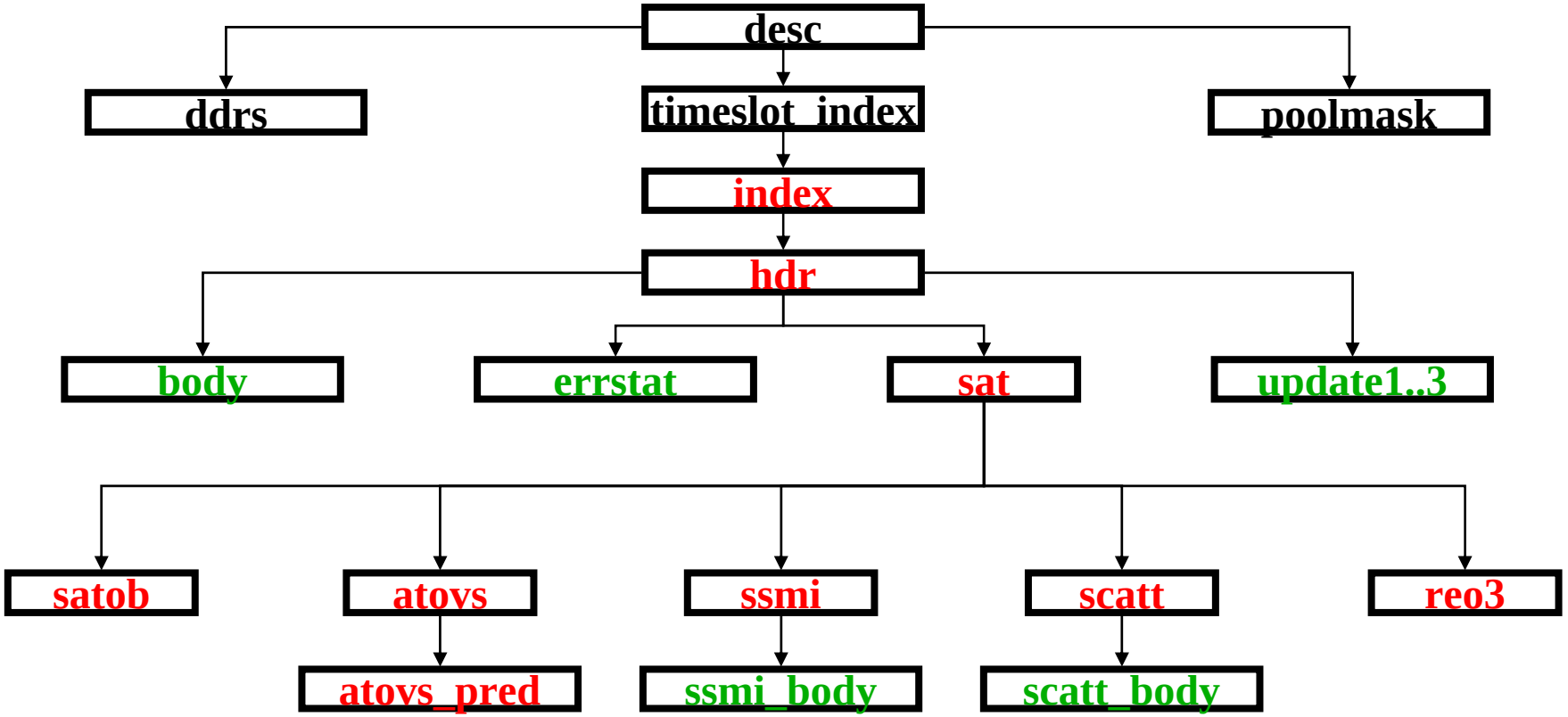
- **Sami Saarinen and al. came up with the idea of using relational database concepts for easier data selection and filtering: the ODB software was born (mid-1998; became operational in 2000).**

- **But what is ODB?**
  - An incore database (like CMA) to improve efficiency
  - A format: inherited from CMA format (hierarchical format)
  - A hierarchical database with a data definition and query language: ODB/SQL – language (subset of ANSI SQL)
  - A parallel fortran 90 interface to enable MPI-parallel data queries, but also to coordinate queries for data shuffling between MPI-tasks
  - A set of post-processing tools (odbsql, odbdiff, etc.)

But ODB cannot...

- **Restrict the user's ability to retrieve, add or modify data by protecting unauthorized access. However, with Fortran90 access layer, an ODB database can be opened in READONLY-mode.**
- **Share a database by concurrent users without interfering each other. Possible for READONLY-databases.**
- **Protect the database from corruption due to inconsistent updates or during system failures.**

ODB hierarchical data model – ECMWF layout



- ***ssmi_body, scatt_body*** **tables are similar to body table**

How to describe this hierarchy?

ODB/SQL: Data Definition Language (DDL)

```
CREATE TABLE hdr AS (
lat real,
lon real,
statid string,
obstype int,
date YYYYMMDD,
time HHMMSS,
status flags_ t,
body @LINK,
);
CREATE TABLE body AS (
varno pk5int,
press pk9real,
obsvalue pk9real,
);
```

lat	lon	statid	obstype	date	time	status
-14.78	143.5	' 94187'	1	20081021	230000	1

@LINK

varno	press	obsvalue
1	100350	804.14
30	100100	120
39	99900	277.6
40	100350	292.4
58	100350	0.57
111	100840	260
112	100100	2
41	97670	12.9
42	95310	-4.84e-15
80	100880	0

A *LINK* tells how many times a row needs to be repeated (10 times in our example) and which table is involved (body)

standard data type

column name or attribute

built-in date & time types

packed data type

composite data type (bit-field)

LINK data type

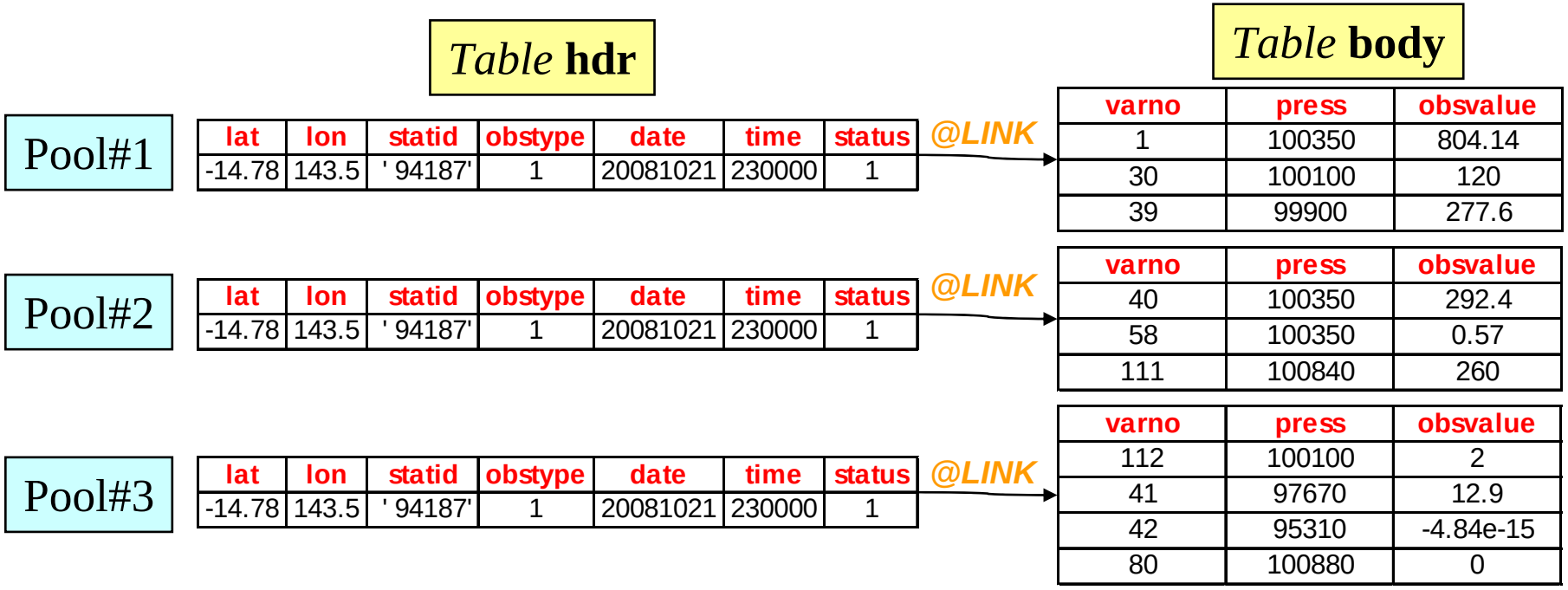
ECMWF

Parallelisation: a requirement for IFS...

ODB parallel database system

- **Aims to improve performance through parallelization of various operations, such as loading data, building ODBs and evaluating queries.**
- **Data is stored in a distributed fashion**
  - divide TABLEs “horizontally” into pools between processors; pools are assigned to the MPI-tasks in a round-robin fashion.
  - each table can be assigned to an openMP threads
- **no. of pools "decided" in the Fortran90 layer**
- **SELECT data from *all* or a *particular* pool only**
- **Distribution of data among pools done at the ODB creation**

Example of data partitioning

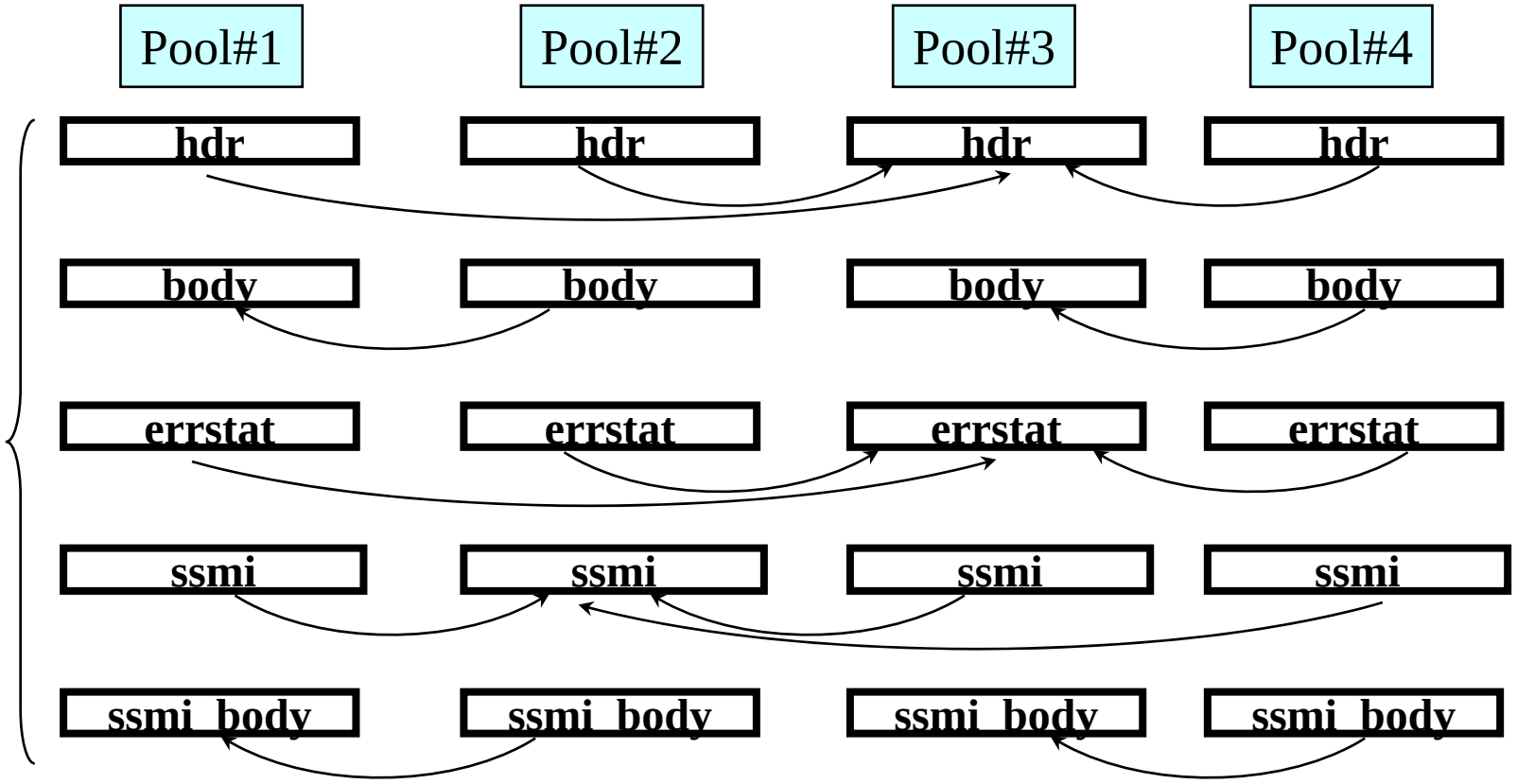


- **A single pool forms a ‘sub-database’.**

Parallel I/O strategy

I/O tasks

- To improve performance, only a subset of pools is selected to perform I/O (read/write ODB on disk). Similar tables are then concatenated together.
- The number of I/O pools is fully configurable



ECMWF

Example of an ODB database on disk

> ls ECMA.iasi

1/	141/	183/	218/	265/	43/	85/
107/	145/	193/	225/	266/	49/	97/
110/	15/	197/	239/	267/	56/	99/
113/	155/	211/	241/	272/	57/	
121/	164/	212/	25/	281/	71/	
127/	169/	217/	253/	29/	73/	

Pool directories

ECMA.iomap
ECMA.sch
IOASSIGN@
ECMA.IOASSIGN
ECMA.dd
ECMA.flags

Metadata

> ls ECMA.iasi/1

atovs	ddrs	index	sat	ssmi	update_2
atovs_body	desc	poolmask	satob	ssmi_body	update_3
atovs_pred	errstat	reo3	scatt	timeslot_index	
body	hdr	reo3_body	scatt_body	update_1	

Data selection and filtering...

→ **To read/update your database once it is created…**

ODB/SQL Queries – For existing ODBs only...

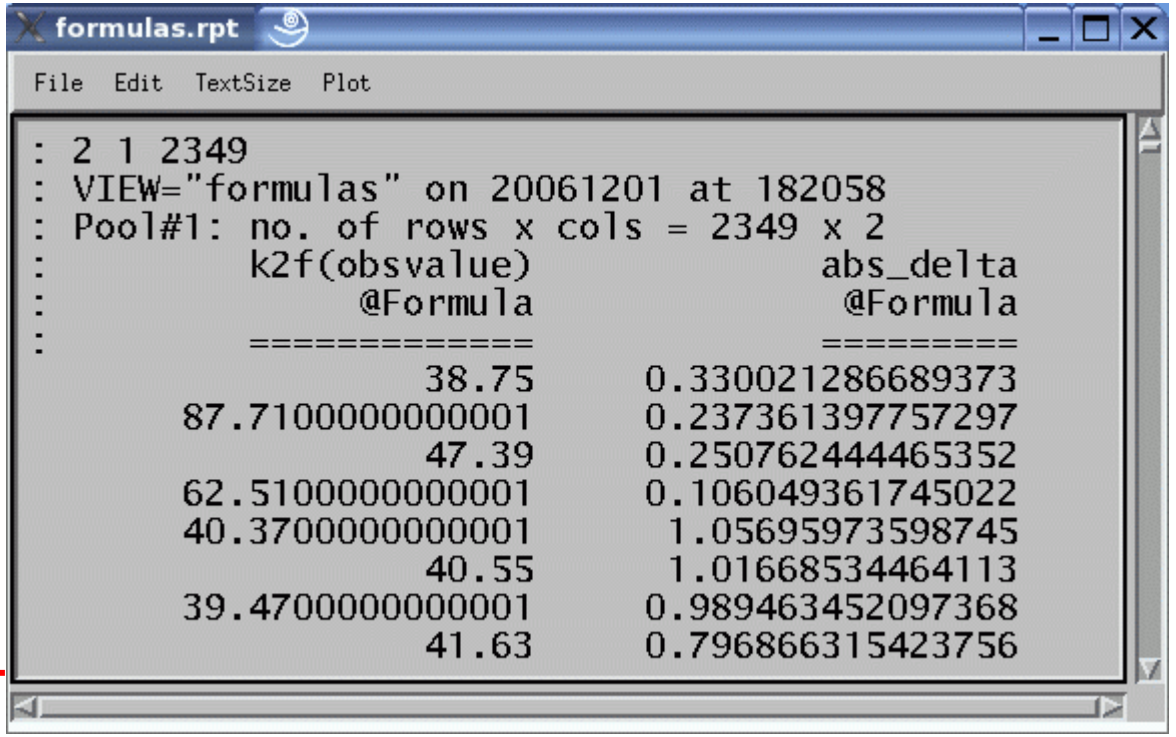
```
[CREATE VIEW view_name AS]
SELECT [DISTINCT] column_ name( s)
FROM table( s)
[WHERE some_ condition( s)_ to_ be_ met ]
[ORDERBY sort_ column_ name( s) [ASC/ DESC] ]
```

- **ODB/SQL(*) is a small subset of international standard SQL used to manipulate relational databases.**
- **It allows to define data queries in order retrieve (in parallel) a subset of data items. This is the “main” motivation of using ODB ?!**
- **Except for the creation of a database or within IFS/ARPEGE where a Fortran program is necessary, ODB/SQL can be used in an interactive way via ODB-tools (odbviewer, odbsql, etc.).**

(*)SQL stands for Structured Query Language

ODB/SQL example

```
SELECT fahrenheit(obsvalue),  // Convert from Kelvin to F
       abs(fg_depar - an_depar) AS abs_delta
FROM hdr, body
WHERE
obstype = $synop
AND
varno@body = $t2m
AND
obsvalue is not NULL
;
```



odbsql -v request.sql -i /home/rd/stf/ECMA.conv

What about parallel data queries?

Fortran 90 interface to ODB/SQL

- **Parallel** data queries are possible via the ODB Fortran90 interface layer;
- The Fortran 90 layer offers a unique user interface to
  - Open & close database
  - execute ODB/SQL queries, ***update*** & store queried data
  - Inquire information about database metadata
- The same code can be used in serial or parallel MPI/OpenMP mode (with any number of processors/openMP threads).
- SELECT‘ ed data can be asked to be **shuffled** (“ part- exchanged”) or **replicated** across processors; by default data selection applies to the **local pools** only.

An example of Fortran program with ODB

```
h = ODB_open("ECMA", "OLD", npools=npools)
DO jp=1,npools
   rc= ODB_select(h, "sqlview",nrows,ncols,poolno=jp)
   allocate(x(nrows,0:ncols))
   rc= ODB_get(h, "sqlview",x,nrows,ncols,poolno=jp)
   call update(x,nrows,ncols) ! Not an ODB-routine
   rc= ODB_put(h, "sqlview",x,nrows,ncols,poolno=jp)
   deallocate(x)
   rc= ODB_cancel(h, "sqlview",poolno=jp)
ENDDO
rc= ODB_close(h, save=.TRUE.)
end program main
```

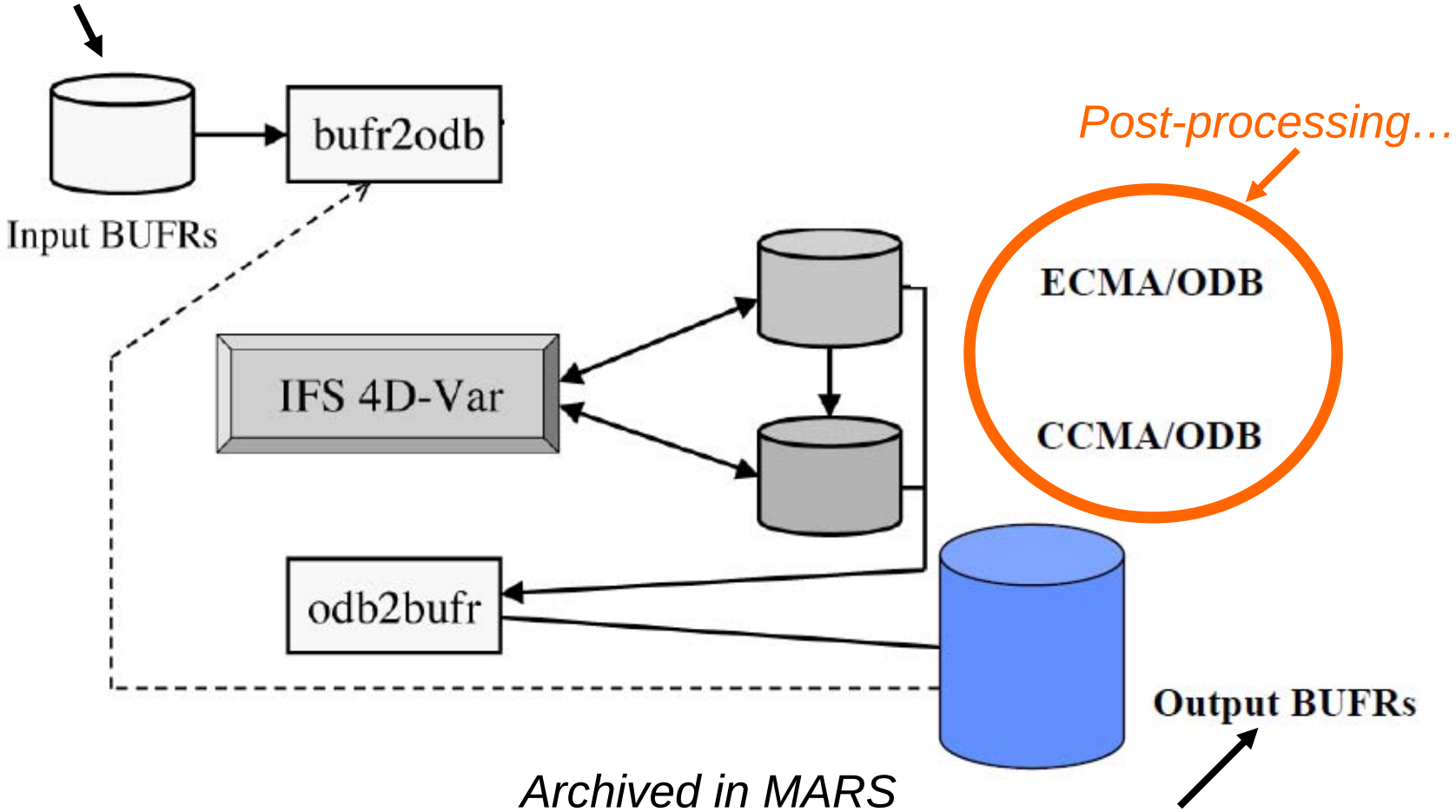
But how does it work in our 4Dvar system?

ECMWF usage of ODB

- **We use two main ODBs:**
  - **ECMA (Extended CMA): all observations *(active/passive/blacklisted)***
  - **CCMA (Compressed CMA): active observations after IFS screening**
- **No unique centralized ODBs: we create new ODBs for each analysis**
- **ECMAs are created from bufr files:**
  - Enables MPI-parallel database creation → efficient
  - Distribution is done in bufr2odb in IFS for ECMA (pools done per obs. group). It is done again when creating CCMA from ECMA i.e. when creating a new database with active data only.
- **ODBs archived in ECFS which is a large distributed storage system**
- **Feedback bufr files are created from ODBS at the end of the analysis and archived in MARS our Meteorological Archive.**

ODB within IFS/4Dvar system

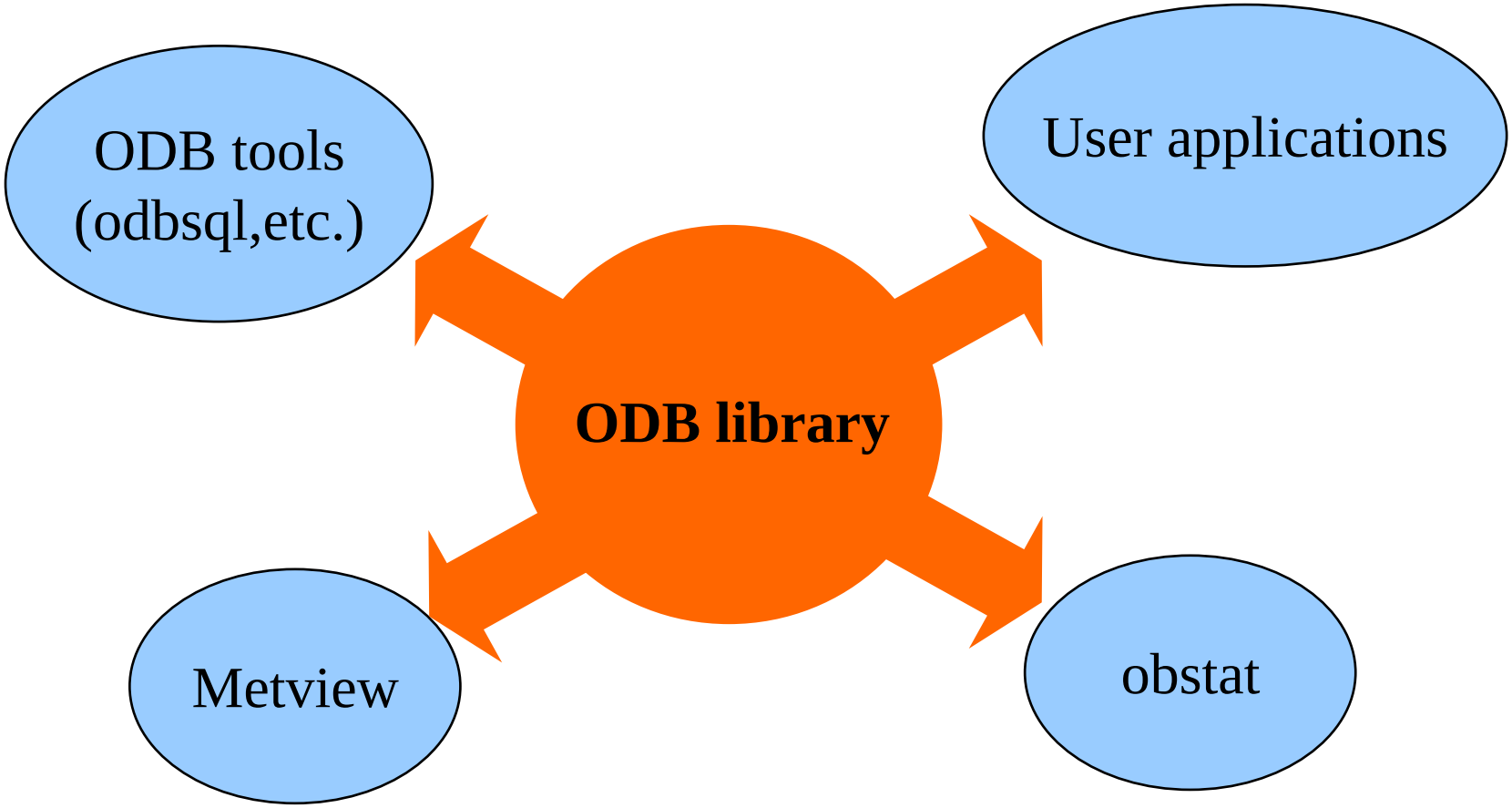
Archived in MARS
or available on line on our HPCF



Archived in MARS
(ECMWF main repository of meteorological data)

Post-processing of ODBs...

ODB-tools and post-processing applications



- **Metview: plotting package (see Sandor presentation done this morning**

The way forward...

What next?

- **ODB is now more than a tool dedicated to our 4Dvar system. It is now time to better integrate ODB in our full ECMWF system (from receiving observations to the archiving of feedback information)**

→ **First step is to archive ODBs in our Meteorological archive (see Peter Kuchta presentation on Friday)**

- **More and more interest on ODB from external centres (ODB used by Australian Bureau of Meteorology, Melbourne; triggered some interest by UK Met Office; GMAO, Washington investigates the possibilities of ODB for their own usage, etc.)**

→ **Make ODB easier to handle by external parties: revisit ECMWF DDL file, create a dictionary of ODB attributes and their usage, improve user interfaces, etc.**