KM3NeT
Opens a new window on our universe

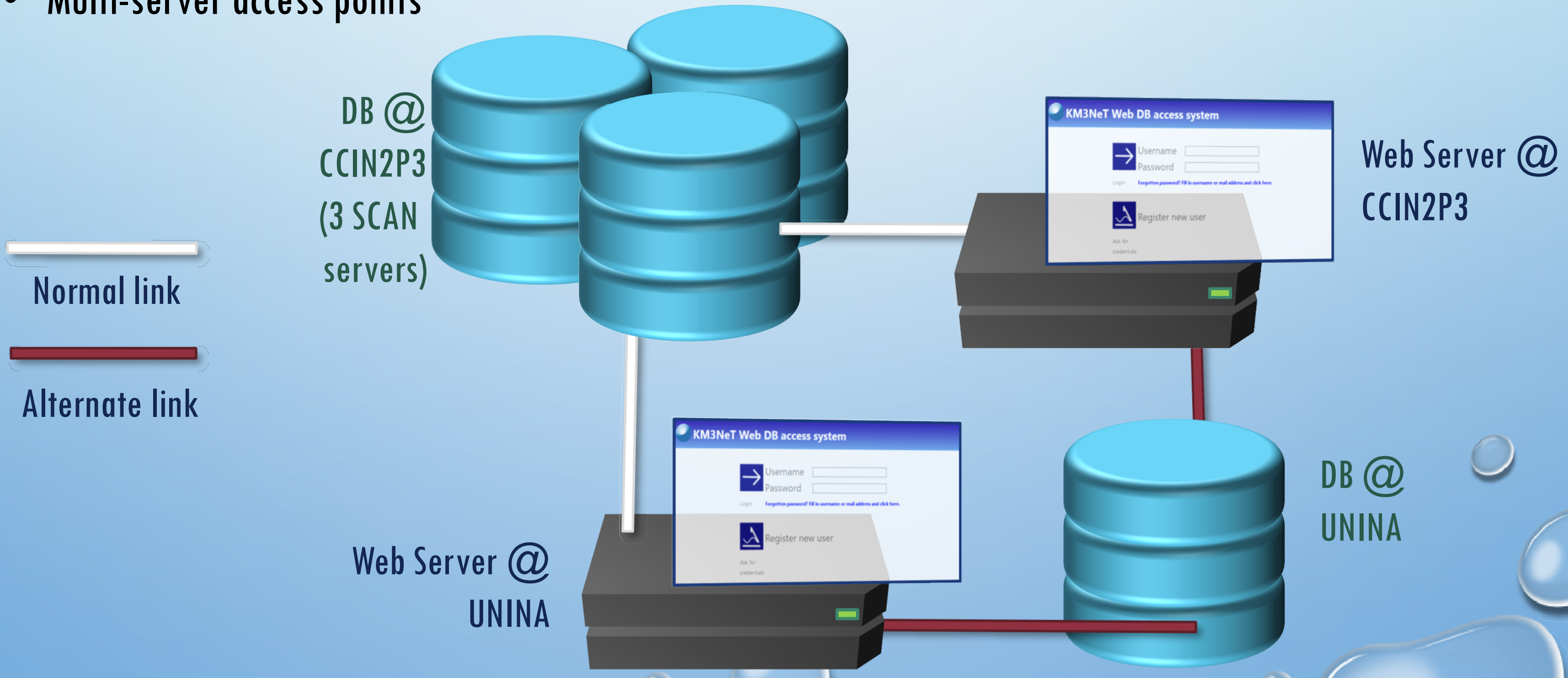
Very Large Volume Neutrino Telescope
VLVνT
workshop

The Relational Database System of KM3NeT

A. Albert, C. Bozza for the KM3NeT Collaboration

Overview

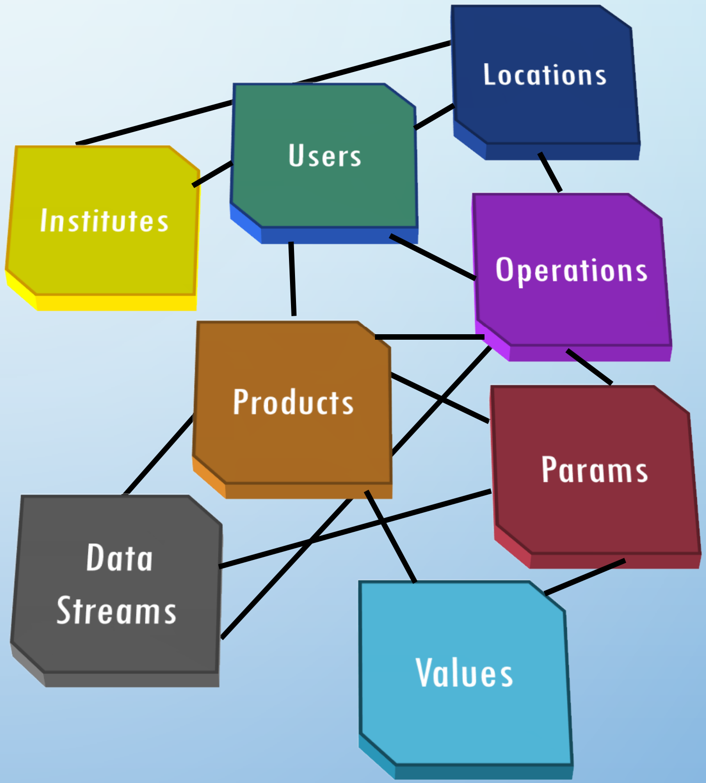
- Many tables and concepts derived from ANTARES Database
- Uses Oracle Database Server Enterprise Edition (currently 12c)
- Geographically distributed system in multi-master configuration
- Multi-server access points



Database schema

- Basic contents:
  - Users
  - Locations (physical or postal addresses)
  - Institutions
  - Operations (simple or complex actions, identified in time, place, responsibility)
  - Products (i.e., HW or SW components)
  - Parameters (calibration/configuration/ characterisation)
  - Values
  - Data Streams

Other aspects (e.g. paper management) present but not covered in this talk

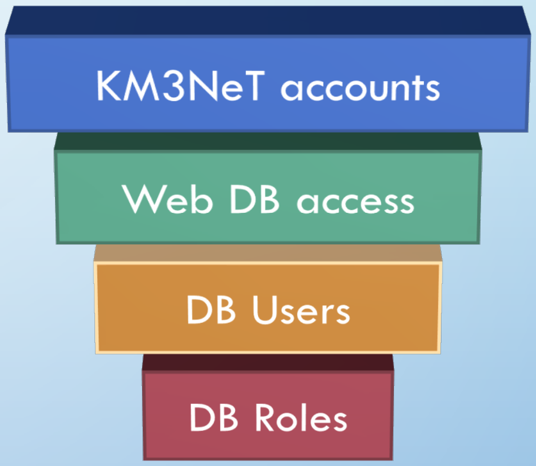


Simplified DB schema – real one has 92 tables

Credential handling and security

- Oracle accounts assigned to instances of IT services
  - In addition: “reader” and “writer” Oracle roles
- KM3NeT user accounts are not Oracle accounts
  - KM3NeT accounts have privileges related to Collaboration-assigned functions and appointments
- User passwords irreversibly encrypted by hashing
- User roles and permissions written in dedicated tables
  - Logical protection layer beyond “read / write” access
- Centralized credentials for all IT services
  - Access to Database Web interface
  - Google Drive
  - Wiki
  - SVN/TRAC servers
  - Collaborative LaTeX
  - …

4 security levels

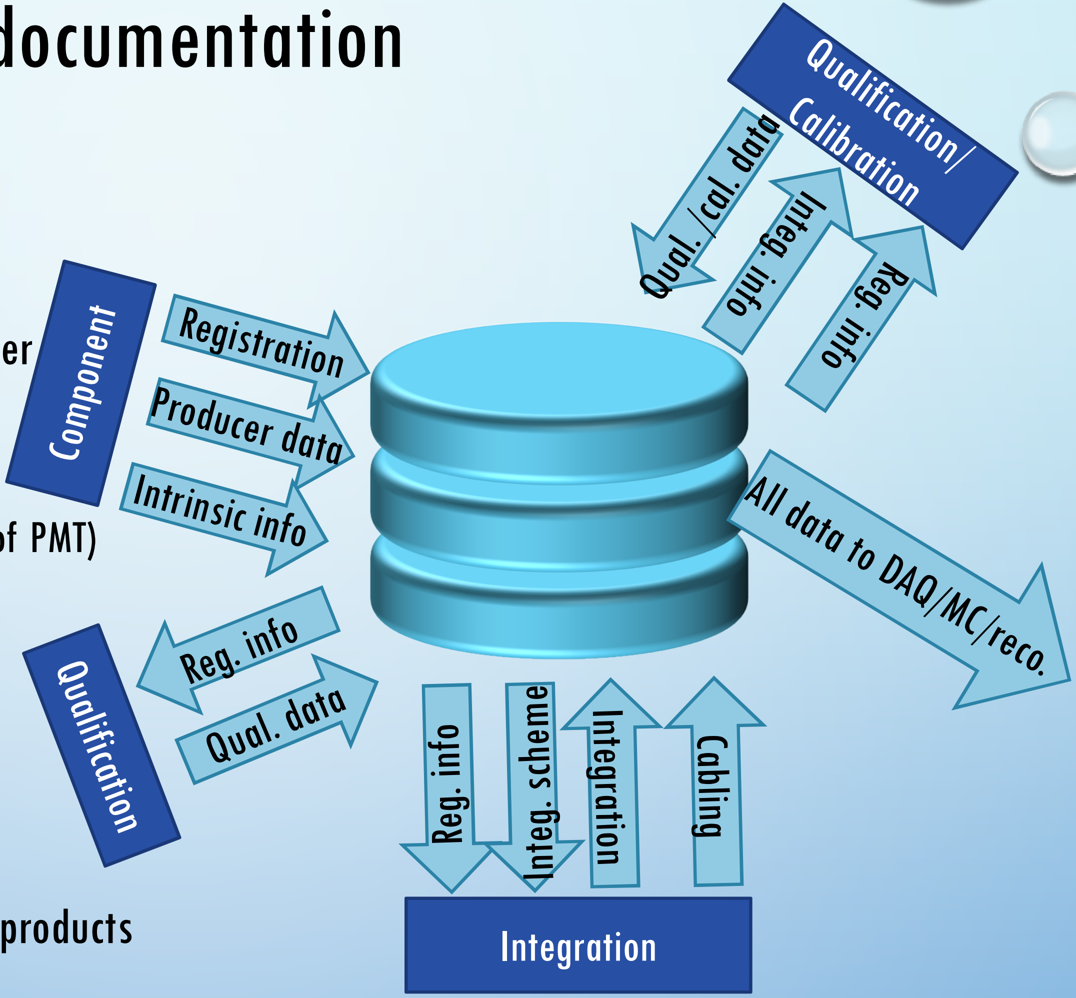


Data flow for construction documentation

- Steps in construction:

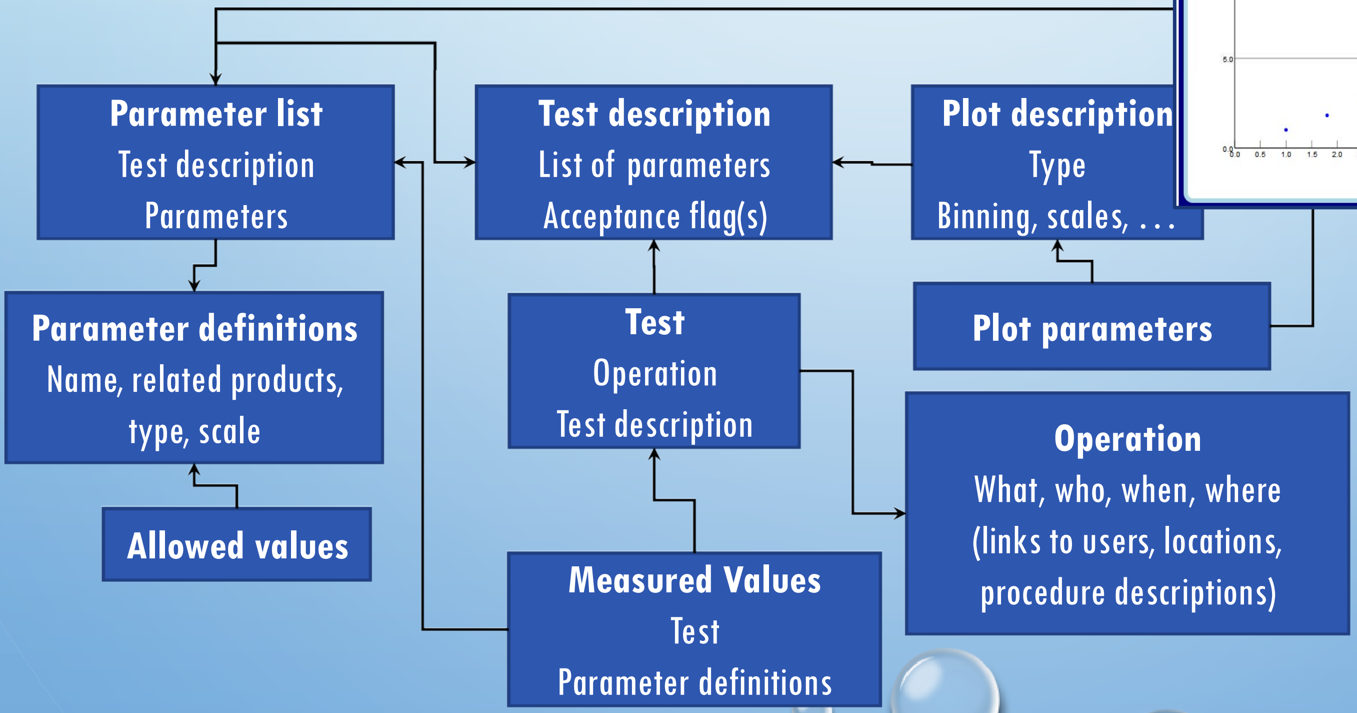
1. Registration of product or container
2. Documentation
   - Producer data (e.g. nominal HV of PMT)
   - Intrinsic info (e.g. MAC address)
3. Testing components
4. Integration
5. Qualifying/calibrating integrated products
6. **Loop from 4**

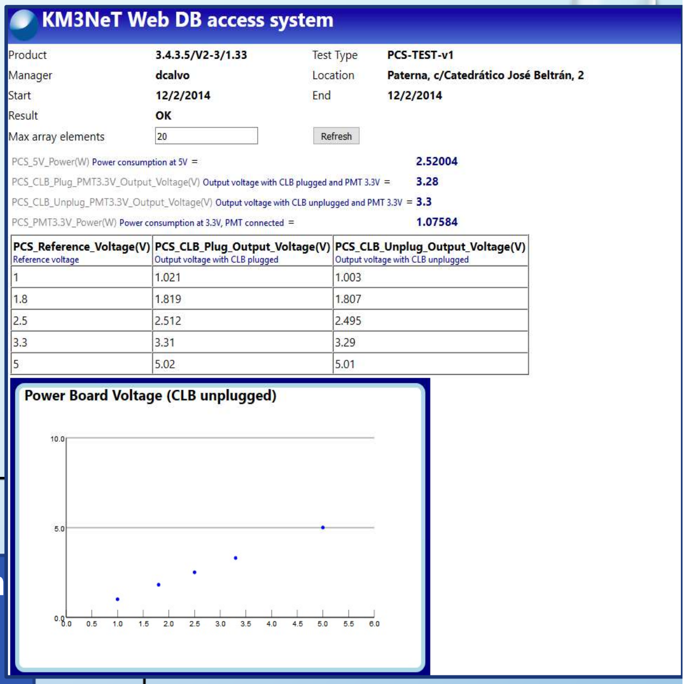
DAQ, simulations and reconstruction/analysis will use all these data
Each step needs data from the previous ones



Data flow for construction documentation

- Qualification and calibration procedures described in DB
- Parameters always in standard units along with "convenient scale" (e.g. mA, nm, etc.)
- Reporting plots defined in DB and generated dynamically




KM3NeT Web DB access system

Product	3.4.3.5/V2-3/1.33	Test Type	PCS-TEST-v1
Manager	dcalvo	Location	Paterna, c/Catedrático José Beltrán, 2
Start	12/2/2014	End	12/2/2014
Result	OK		
Max array elements	20	Refresh	

PCS_5V_Power(W) Power consumption at 5V = 2.52004
PCS_CLB_Plug_PMT3.3V_Output_Voltage(V) Output voltage with CLB plugged and PMT 3.3V = 3.28
PCS_CLB_Unplug_PMT3.3V_Output_Voltage(V) Output voltage with CLB unplugged and PMT 3.3V = 3.3
PCS_PMT3.3V_Power(W) Power consumption at 3.3V, PMT connected = 1.07584

PCS_Reference_Voltage(V) Reference voltage	PCS_CLB_Plug_Output_Voltage(V) Output voltage with CLB plugged	PCS_CLB_Unplug_Output_Voltage(V) Output voltage with CLB unplugged
1	1.021	1.003
1.8	1.819	1.807
2.5	2.512	2.495
3.3	3.31	3.29
5	5.02	5.01

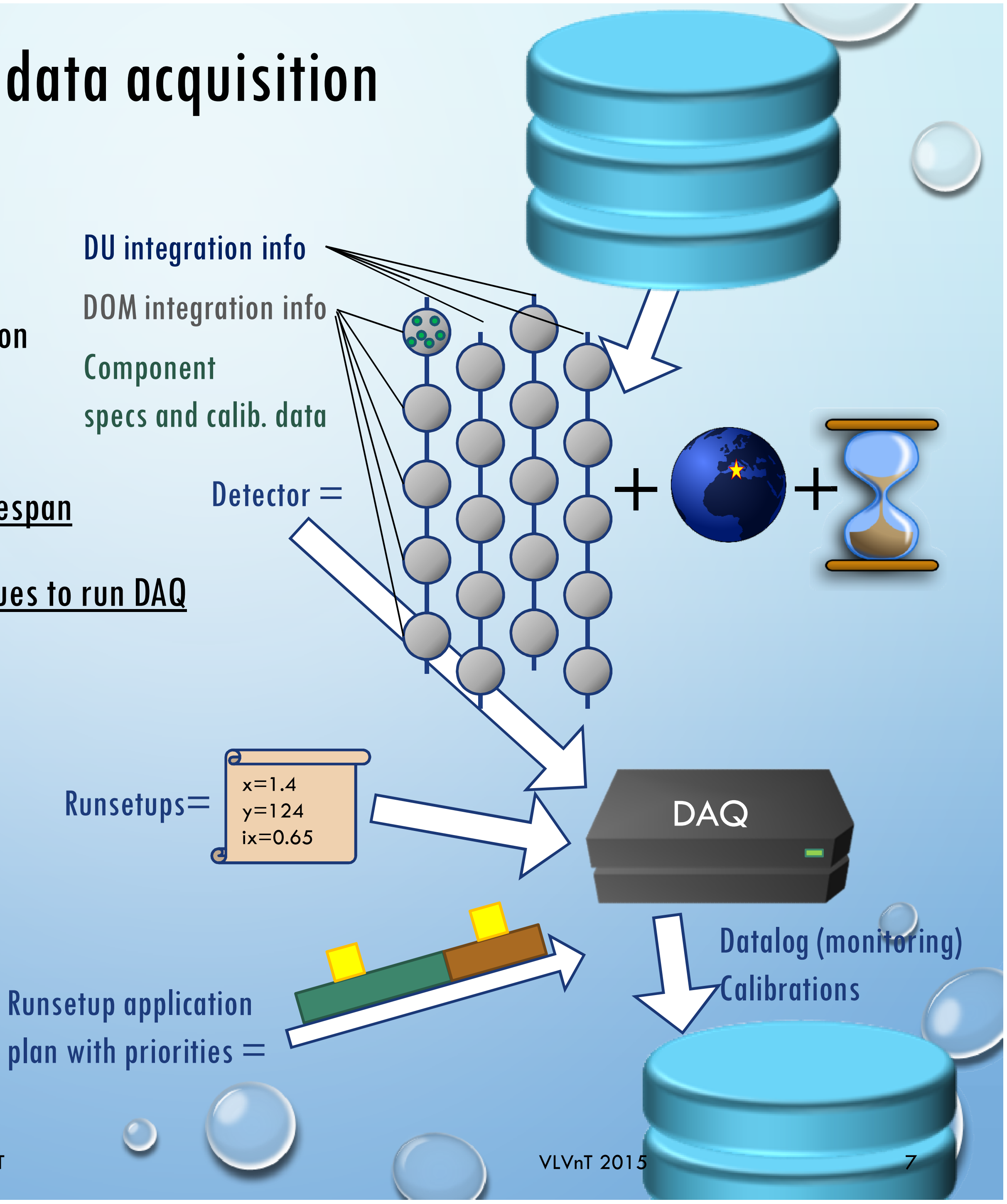
Data flow for support to data acquisition

- Data acquisition possibly uses
  - all info from detector construction
  - DAQ-specific info

Detector = DU's + location + timespan

Runsetup = control parameter values to run DAQ

Runsetups are scheduled using application plans with priorities



Database schema

- Performance vs. ease of management

Non-critical objects: generation rate < 0.001 Hz ($\sim 3 \times 10^5$ in 10 years)

- Users
- Locations
- Products (components)
- Detectors
- Runsetups (DAQ running parameter sets)
- Data-taking runs

- Table storage: plain
- Primary keys generated by single DB-Instance-wide char sequence **A00000001-A00000002…**

Critical objects: generation rate ≥ 0.001 Hz

- Slow control data samples
  - 0.01 Hz $\times$ 2070 DOMs = 20.7 Hz for 115 DUs
  - 0.01 Hz $\times$ 2070 DOMs $\times$ 31 PMTs/DOM = 0.64 kHz for 115 DUs
- Acoustic data samples
- Positional configurations and Calibrations
- Overall estimate: 0.1~1 TB/y

- Index-Organized Tables
- Composite primary keys
- Columnar DB
- Stream-like structure

Database schema

- Performance and ease of management
  - Plain storage
    - (Composite) primary key in plain table
    - Key fields data repeated in the key index
    - Intuitive, easy to understand
    - Fixed set of fields (adding columns to table with 10^9 rows is unrealistic)
    - Data fields relatively compact, but fields can't be skipped in data retrieval

Key field #1	Key field #2	Data field #1	Data field #2	Data field #3
Key field #1	Key field #2	Data field #1	Data field #2	Data field #3
	⋮			
Key field #1	Key field #2	Data field #1	Data field #2	Data field #3

Database schema

- Performance and ease of management
  - Columnar data storage with plain table
    - Composite primary key in plain table
    - Table data repeated in the key index
    - Disk occupancy for N records: 84N (table) +80N (key) =164 N bytes

Datalog ID: 12 bytes	Source Name (32 chars)	Parameter Name: 32 chars	Timecode (4 bytes)	Data (4 bytes)
Datalog ID: 12 bytes	Source Name (32 chars)	Parameter Name: 32 chars	Timecode (4 bytes)	Data (4 bytes)
	⋮			
Datalog ID: 12 bytes	Source Name (32 chars)	Parameter Name: 32 chars	Timecode (4 bytes)	Data (4 bytes)

Database schema

- Performance and ease of management

  Storage for critical objects: e.g. DWORD slow control stream

  Composite primary key in Index-Organized-Table

  Table data stored in the key

  Non-varying fields are not repeated

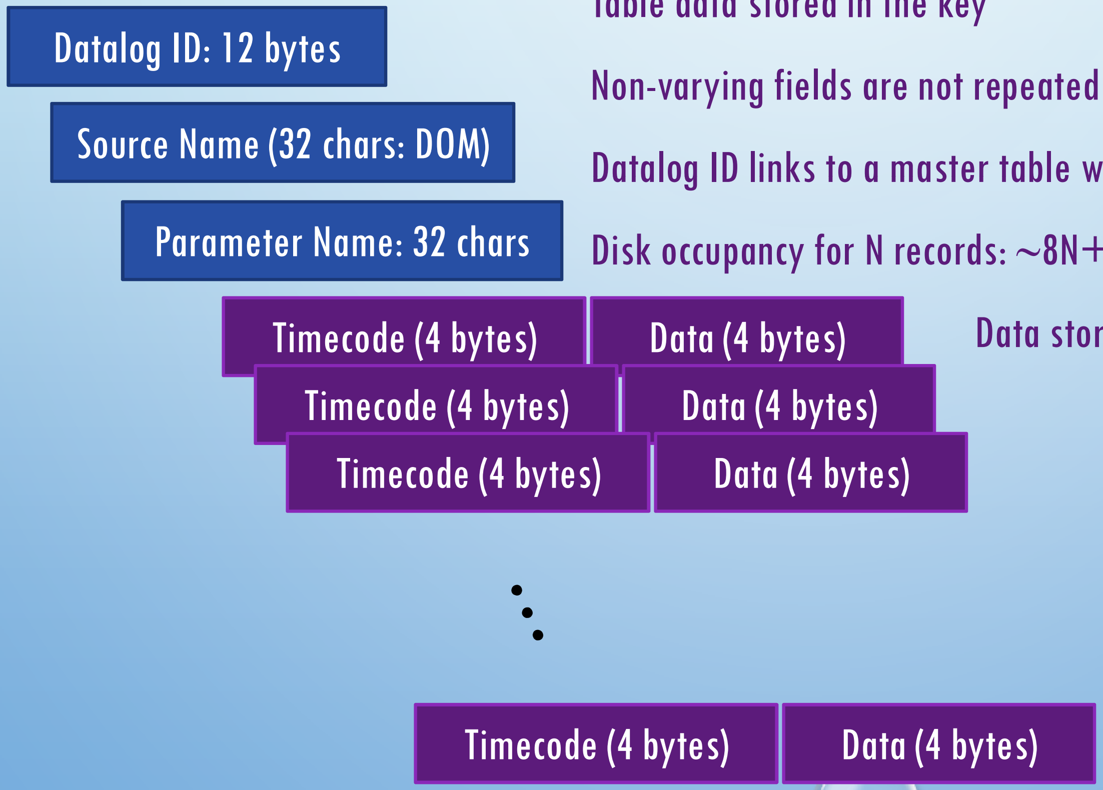
  Datalog ID links to a master table where the time base is defined

  Disk occupancy for N records: ~8N+76 bytes~8 N bytes

  Data storage shrinkage by 164/8=20.5

  No data compression!

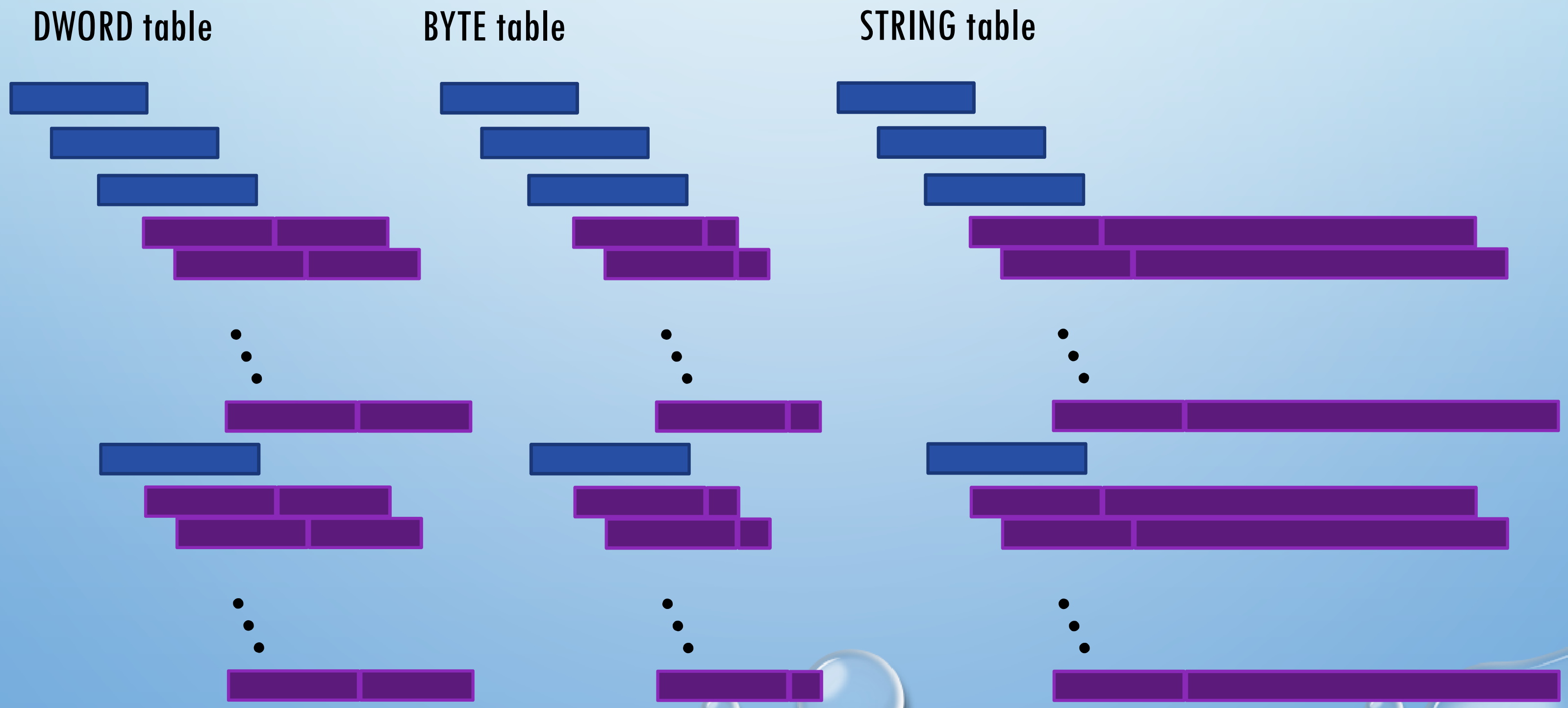
  Access is faster!



Database schema

"Columnar" DB

- Each data type has a different table
- Source selection occurs in the key, not in the table name
- Data retrieval improvement: skip access to unwanted fields (key portions in the tables)
- Define new parameters by just adding entries in the key

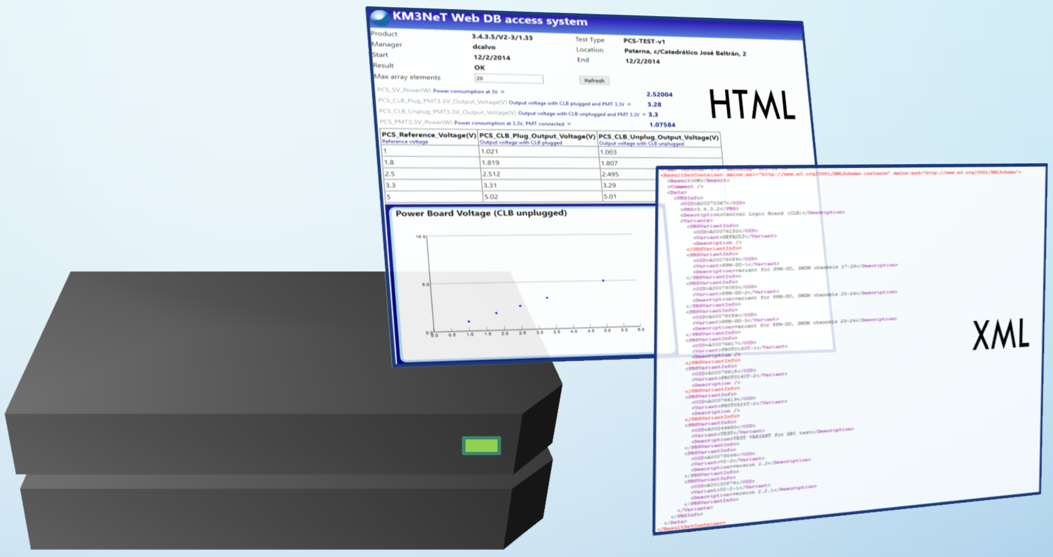


Access through Web

- Application Web Server
  - Provides GUI
  - Provides data access in various formats
  - Can cache data and optimize DB connection usage
  - Runs optimized SQL
  - Queues long-running data management jobs
  - Prevents problems (e.g., table locking due to incomplete writes in case of broken connections)
  - No Oracle client needed: read/write data by browser, *wget*, *curl* or any HTTP communication library
  - Compatible with GRID computing
- Developed on top of the Common Language Runtime standard (ECMA)
  - Written in C#
  - Runs in Mono, .NET, DotGNU on Linux, Windows, Mac OS (binary compatible!)
  - High performance thanks to JIT compilation
  - Lightweight (400 kB binaries, 12 MB including libraries and HTML)
  - Garbage collection, LINQ, Lambda calculus help avoiding fatal bugs (crashes, service outage)

Access through Web

- HTTP(S) supported
- GUI, user-oriented HTML pages
- Data access service:
  - ASCII (JPP)
  - XML
  - JSON
  - Possible: ROOT file encoding (under evaluation)
- Managing insertion of large datasets
  - Direct DB insertion (HTTP answer waits for DB insert completion) for small datasets
  - Batch insertion recommended for GB-size datasets (HTTP request would timeout on most clients)
    - Data transferred to staging area
    - User receives email report on completion/failure



KM3NeT Web DB access system

Product	3.4.3.5/V2-3/1.33	Test Type	PCS-TEST-v1
Manager	dcalvo	Location	Paterna, c/Catedrático José Beltrán, 2
Start	12/2/2014	End	12/2/2014
Result	OK		
Max array elements	20		

PCS_5V_Power(W) Power consumption at 5V = 2.52004
PCS_CLB_Plug_PMT3.3V_Output_Voltage(V) Output voltage with CLB plugged and PMT 3.3V = 3.28
PCS_CLB_Unplug_PMT3.3V_Output_Voltage(V) Output voltage with CLB unplugged and PMT 3.3V = 3.3
PCS_PMT3.3V_Power(W) Power consumption at 3.3V, PMT connected = 1.07584

PCS_Reference_Voltage(V) Reference voltage	PCS_CLB_Plug_Output_Voltage(V) Output voltage with CLB plugged	PCS_CLB_Unplug_Output_Voltage(V) Output voltage with CLB unplugged
1	1.021	1.003
1.8	1.819	1.807
2.5	2.512	2.495
3.3	3.31	3.29
5	5.02	5.01

```
<ResultSetContainer xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Result>OK</Result>
  <Comment />
  <Data>
    <PBSInfo>
      <OID>A00075367</OID>
      <PBS>3.4.3.2</PBS>
      <Description>Central Logic Board (CLB)</Description>
      <Variants>
        <PBSVariantInfo>
          <OID>A00076182</OID>
          <Variant>DEFAULT</Variant>
          <Description />
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00078084</OID>
          <Variant>PPM-DU-1</Variant>
          <Description>variant for PPM-DU, DWDM channels 27-28</Description>
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00078085</OID>
          <Variant>PPM-DU-2</Variant>
          <Description>variant for PPM-DU, DWDM channels 25-26</Description>
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00078086</OID>
          <Variant>PPM-DU-3</Variant>
          <Description>variant for PPM-DU, DWDM channels 23-24</Description>
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00076617</OID>
          <Variant>PROTO160T-1</Variant>
          <Description />
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00076618</OID>
          <Variant>PROTO160T-2</Variant>
          <Description />
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00076619</OID>
          <Variant>PROTO325T-2</Variant>
          <Description />
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00249660</OID>
          <Variant>TEST</Variant>
          <Description>TEST VARIANT for DBI test</Description>
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00079546</OID>
          <Variant>V2-2</Variant>
          <Description>version 2.2</Description>
        </PBSVariantInfo>
        <PBSVariantInfo>
          <OID>A00100876</OID>
          <Variant>V2-2-1</Variant>
          <Description>version 2.2.1</Description>
        </PBSVariantInfo>
      </Variants>
    </PBSInfo>
  </Data>
</ResultSetContainer>
```

Access through Web

XML/JSON Data services

- Complex data types: parameters, run setups, detectors, etc.
- For each data type, define:
  - Data class (if not already defined elsewhere)
  - SELECT, INSERT, UPDATE, DELETE strings with SQL commands
  - Query restriction fields for SELECT (selector fields)
  - Methods to convert result set row to/from "data class"
  - Mono/.NET provide automatic serialization support for any reasonable "data class" to represent it on the HTTP stream both in XML and JSON – No need to write serialization code
  - The main page also provides online documentation (automatically generated) on available methods and selector fields

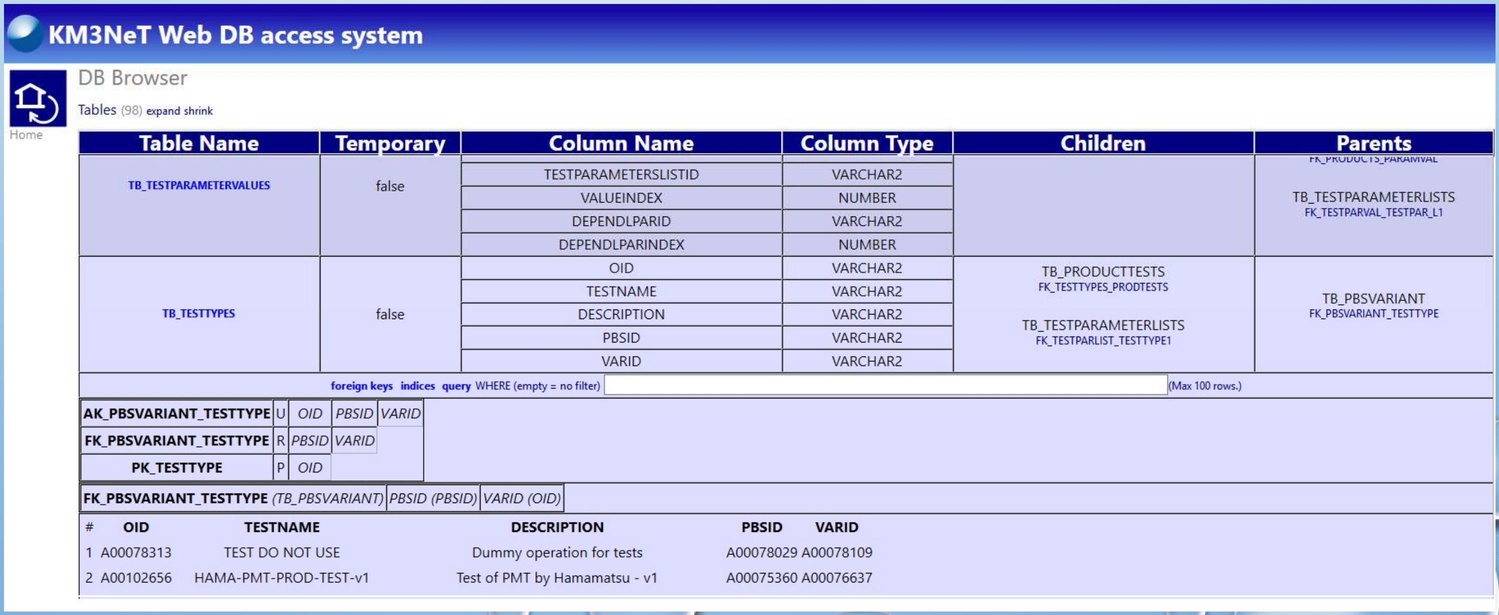
Examples:

http://myserv.er/xmlds/runsetups/select?oid=R03201

http://myserv.er/jsonds/runsetups/insert (+file sent via POST)

Access through Web

- HTML Database browser
  - Restricted to administrators and a few more qualified users
  - Dynamically browse all tables (uses no *a priori* knowledge of the DB schema)
  - Make queries
  - Follow links (foreign keys) back and forth



KM3NeT Web DB access system

Home

DB Browser

Tables (98) expand shrink

Table Name	Temporary	Column Name	Column Type	Children	Parents
TB_TESTPARAMETERVALUES	false	TESTPARAMETERSLISTID	VARCHAR2		FK_PRODUCTS_PARAMVAL TB_TESTPARAMETERLISTS FK_TESTPARVAL_TESTPAR_L1
		VALUEINDEX	NUMBER		
		DEPENDLPARID	VARCHAR2		
		DEPENDLPARINDEX	NUMBER		
TB_TESTTYPES	false	OID	VARCHAR2	TB_PRODUCTTESTS FK_TESTTYPES_PRODTESTS TB_TESTPARAMETERLISTS FK_TESTPARLIST_TESTTYPE1	TB_PBSVARIANT FK_PBSVARIANT_TESTTYPE
		TESTNAME	VARCHAR2		
		DESCRIPTION	VARCHAR2		
		PBSID	VARCHAR2		
		VARID	VARCHAR2		

foreign keys **indices** **query** WHERE (empty = no filter) (Max 100 rows.)

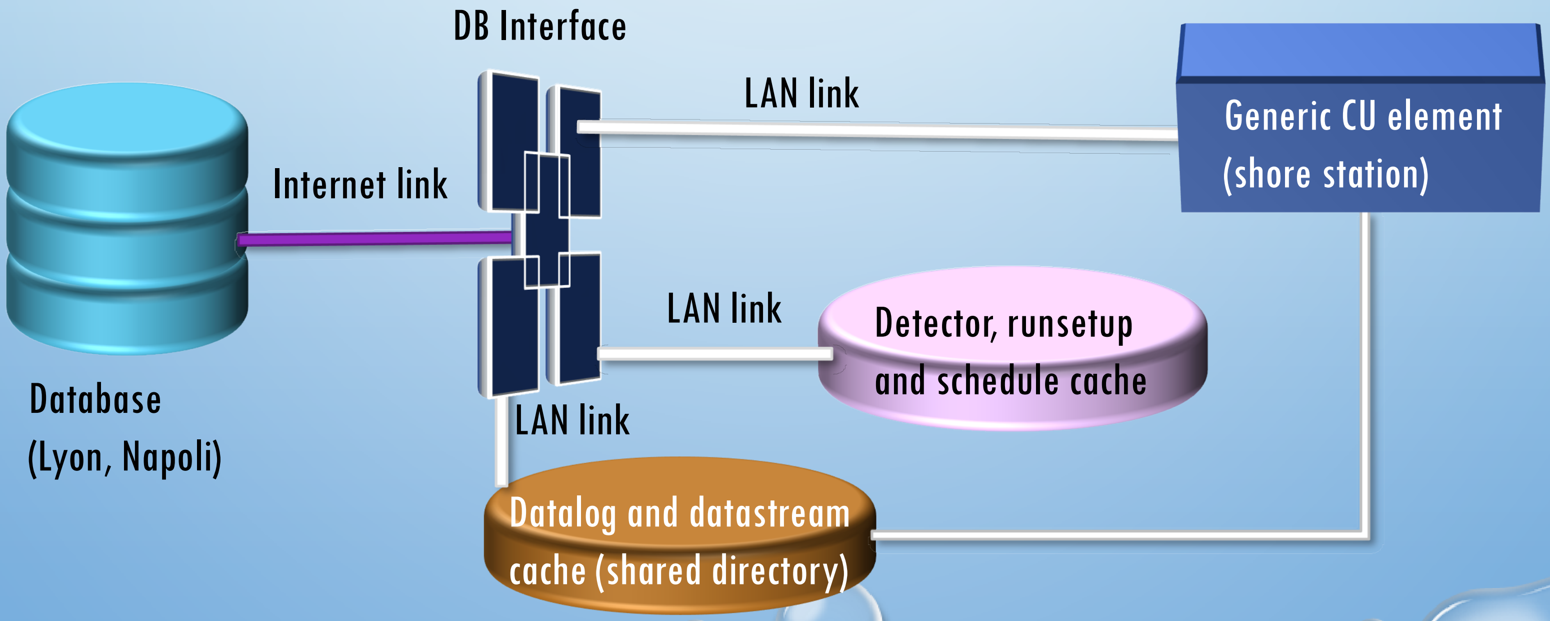
AK_PBSVARIANT_TESTTYPE	U	*OID*	*PBSID*	*VARID*
FK_PBSVARIANT_TESTTYPE	R	*PBSID*	*VARID*	
PK_TESTTYPE	P	*OID*		

FK_PBSVARIANT_TESTTYPE *(TB_PBSVARIANT)* | *PBSID (PBSID)* | *VARID (OID)*

#	OID	TESTNAME	DESCRIPTION	PBSID	VARID
1	A00078313	TEST DO NOT USE	Dummy operation for tests	A00078029	A00078109
2	A00102656	HAMA-PMT-PROD-TEST-v1	Test of PMT by Hamamatsu - v1	A00075360	A00076637

Database access from the shore station

- Database Interface service included in the design of the Control Unit (C. Bozza's talk in session E)
  - Built on top of the same technology as the KM3NeT Application Web Server
  - Adds local caches for specific data types (runsetups, detector definitions, schedules, datalogs, acoustic/calibration data streams) to allow continuous operation even in case of Internet connection loss


DB Interface
LAN link
Generic CU element (shore station)
Internet link
LAN link
Detector, runsetup and schedule cache
Database (Lyon, Napoli)
LAN link
Datalog and datastream cache (shared directory)

Conclusions

Relational database for KM3NeT:

- Not a passive archive but a living system taking active part into detector construction and operation
- Careful design of data storage logic can lead to dramatic performance improvements
- HTTP (Web) access simplifies connection while adding security, flexibility and performance to relational database
- User-friendly interface and self-documented data access services are provided by a lightweight Web server developed on purpose