The MDC accounts in the Oracle database DB-HADES

Status report by Ilse Koenig

GSI, May 2009

Table of Contents

1 Overview

Directly related to MDC are actually four accounts in Oracle:

Account name	*Description*
MDC_ANA	Production account This account contains almost all tables, views and code. Only a few people know the password.
MDC_WWW	Account for MDC specific WebDB developments This account contains only a few packages generated with the WebDB developer GUI. These packages generate forms and reports accessible on the WebDB Site.
MDC_OPER	MDC specific operator account This account has no tables or views, but is purely used to insert data via analysis macros and to manipulate the data via secure and tested applications.
MDC_ANATEST	Test account for new developments (or novice developers) Although the account was once filled via an export of the production account, it not guaranteed, that the design is still identical and complete.

2 The production account MDC_ANA

2.1 Content

Object type	*Number of objects*	*Miscellaneous Details*
Tables	68 (+1 backup table)	425 columns, 212 constraints, 102 indexes, 14 triggers
Views	60	521 columns
Packages	23	7873 lines of code
Sequences	6	

The tables may be grouped in seven blocks:

1. the MDC modules, the hardware and the cabling related to the lookup TDC – wire
2. the daughterboards and ADD8 chips
3. the hardware and cabling of the readout
4. readout parameters (thresholds and channel masks)
5. the analysis parameters
6. utility tables used by the WebDB GUIs
7. some obsolete tables (not removed to be backward compatible with old analysis code)

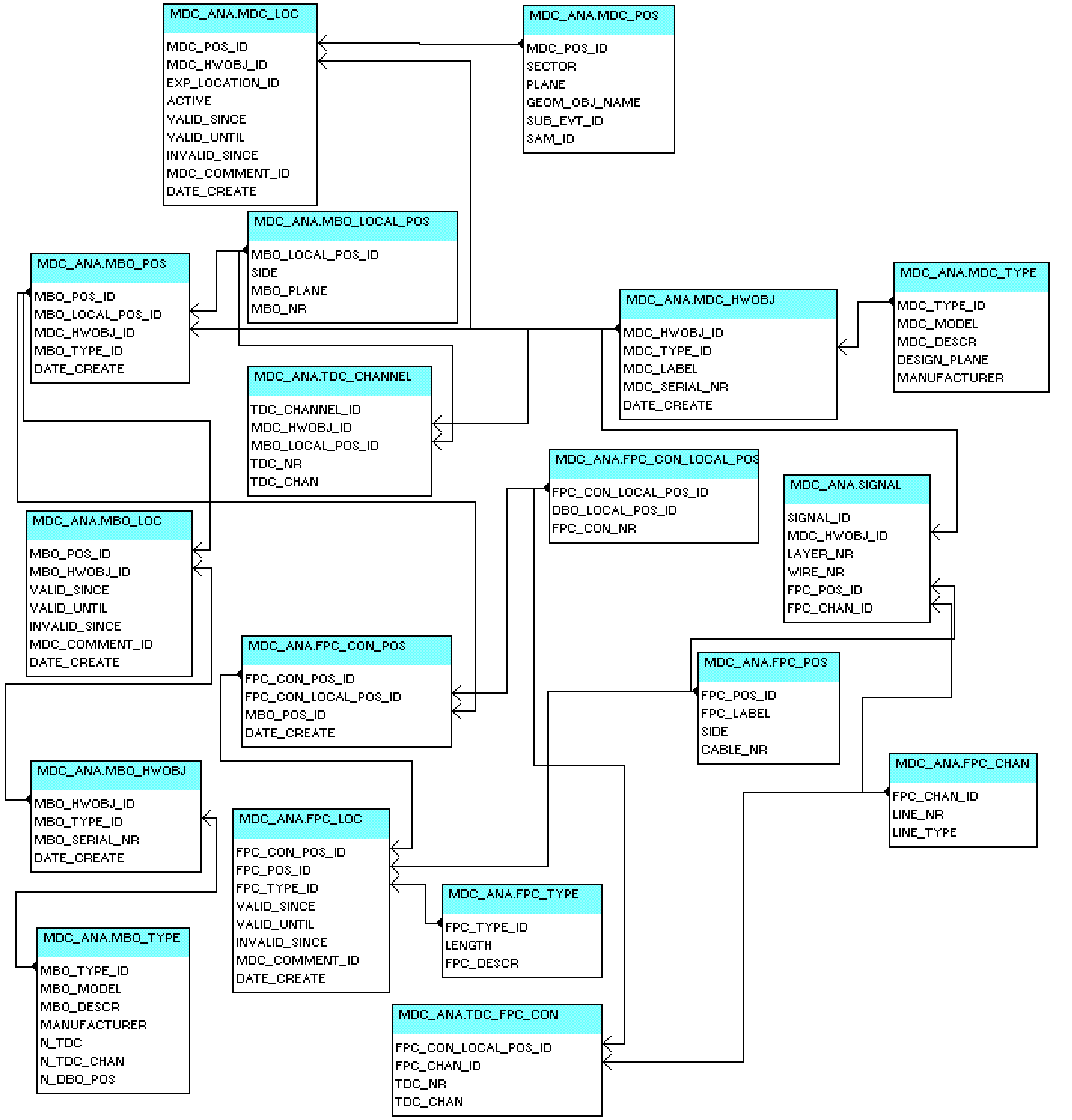
2.2 MDC modules and the cabling TDC - wire

MDC_COMMENT
Comments for changes in MDC tables
All comments in the tables with version managements are stored in this table. The primary key mdc_comment_id is automatically generated by a trigger using the sequence mdc_comment_sequ.

MDC_TYPE
Description of MDC type
This table contains the 4 MDC types.

MDC_HWOBJ
List of MDCs (real detectors, 1..24 +spare)
This table contains actually 26 modules. Originally the primary key corresponded to the label, but the rule was broken by introducing spare modules with ids 25(plane 1) and 26 (plane 2) and by “renaming” some modules at a later stage.
Only the label is actually stored, the serial numbers are still empty.

MDC_POS
List of possible MDC positions
This table contains the 24 possible positions for the MDC modules. The primary key mdc_pos_id is coded as (sector*10 + plane) and ranges from 11 to 64. To each position corresponds a name of the geometry object needed by the geometry containers.



MDC_LOC

Position of MDC objects

This table stores the information, which MDC module is mounted at which position at a certain date. Additionally to the four date columns, the version management distinguishes for the experiment location (e.g HADES_CAVE and VIRTUAL for the simulation).

Since this table needs to be changed only when chambers are mounted or replaced (well known before a beam time), a GUI was never developed. The changes must be done with SQL.

SIGNAL
List of all signal wires (cells)
This table contains the connection between a wire of a module and one of the four signal lines of an FPC cable.

MBO_TYPE
Types of motherboards
This table has actually 2 entries for the short and long MBOs.

MBO_HWOBJ
List of motherboards (hardware)
This table contains all real existing MBOs (mounted plus spare) and additionally some dummy MBOs (label starting with LX and SX) to be used when the label of the mounted MBO is not known.
All short MBOs have labels between MB 0001 and MB 0301, the corresponding primary keys range from S001 to S301. The labels of the long MBOs range from MB 0400 to MB 0563 and the corresponding primary key from L001 to L163.
Eventually not all MBOs manufacted are stored in this table. Missing ones must be inserted with SQL.

MBO_LOCAL_POS
Description of MBO positions in the scope of a chamber
This table contains the 28 possible logical positions on the MDCs.

MBO_POS
Positions of the motherboards on all MDCs
This table contains all possible MBO positions on the 26 MDC modules already manufactured.
The primary key mdc_pos_id is coded as mdc_hwobj_id*10000 + mdc_local_pos_id.

MBO_LOC
Placement of motherboards on motherboard positions
In this table is stored, which MBO is mounted on a MDC module at a certain date.
A trigger firing on insert supports the version management and checks, that the entry is unique (only on object at the specified position, MBO not mounted at more than one position). It also checks, if the MBO type (short or long) is correct for this position.
In case the label of the mounted MBO is not known, a dummy MBO must be mounted at this position. Without this entry, the lookup table for the MDC calibrater would not contain the corresponding channels.
A WebDB GUI allows to insert and change the last actual valid entries. Historic changes (before the last change of the table) must be done with SQL by an expert.

TDC_CHANNEL
List of all TDC channels
This table contains all possible TDC channels on the 26 MDC already manufactured.

TDC_FPC_CON
Link between FPC connectors and TDC channels on a motherboard
This table connects a FPC cable connector pin to a TDC channel in the scope of a MBO.

FPC_TYPE
Type of FPC (FlexPrintCables)
The original aim for the table was to provide the actual cable lengths to the analysis. Since the analysis does not use this information, the table was never filled seriously (only 5 different types).

FPC_CHAN
Channels on FPC cable
This table contains the 10 lines of an FPC cable. fpc_chan_id 1..4 are the 4 signal lines.

FPC_POS
List of FPC (FlexPrintCables) connected to the MDCs
This table contains all FPC cables in use. The primary key fpc_pos_id represents the label (the separate column label is empty!).
The fact, that the FPC type (different cable lengths) is not stored here, but in table fpc_loc) indicates, that this table is more a list of logical objects, than real cables with a label and a fixed length.

FPC_CON_LOCAL_POS
Position of FPC connectors on the six possible daughterboard positions on an MBO (scope of a MBO)
This tables contains the possible 24 connectors on a MBO (4 connectors on each DBO, maximum of 6 DBOs on a (long) MBO).

FPC_CON_POS
Positions of all FPC connectors
This table contains the connectors on all modules and combines the connectors on a MBO with all logical MBO positions.

FPC_LOC
Placement of FPC cables
An error in this table (typically a swap of two cables) results in a wrong lookup table for the MDC calibrater (in this example for 8 channels). Any changes must be done by an expert via SQL, a GUI does not exists.
A trigger firing on insert supports the version management and checks, that the entry is unique (only one FPC object at the specified position, same FPC not mounted at more than one position).
Although each FPC cable has a certain type, it is not guaranteed, that the types (different cable lengths) are correct.

Related views:

View name	*Description*
FPC_LOC_AT_DATE	Access to all actual valid information about FPC cables (label, length, mdc, mbo and dbo connector, layer, side, ...)
FPC_LOC_HISTORY	Same at FPC_LOC_AT_DATE, but for historic data
FPC_LOC_AT_HISTDATE	Historic FPC connector position
MBO_LOC_AT_DATE	Access to all actual valid MBO information (id, label, serial number, position)

View name	*Description*
MBO_LOC_HISTORY	Same at MBO_LOC_AT_DATE, but for historic data
MDCLOOKUPGEOM_AT_DATE	Lookup table for the HYDRA MDC calibrator at a specified date
MDCLOOKUPGEOM_AT_HISTDATE	Same at MDCLOOKUP_AT_DATE, but for a specified history date
RAW_STRUCT_AT_DATE	Number of TDC channels for the connected motherboards at a specified date
RAW_STRUCT_AT_HISTDATE	Same as RAW_STRUCT_AT_DATE, but for a specified history date
MDCTDCNUM_NOW	Number of TDC channels for the actually connected motherboards
MDC_LOC_AT_DATE	Mounted MDCs in the HADES cave at a specified position
MDC_LOC_AT_RUN	Mounted MDCs at a specified run
MDC_LOC_AT_RUN_HIST	Mounted MDCs at a specified run and history date
MDC_LOC_HISTORY	Valid data of mounted MDC modules (label, position, time range)
TDC_NOCON_AT_DATE	Wires not connected to a TDC at a specified date

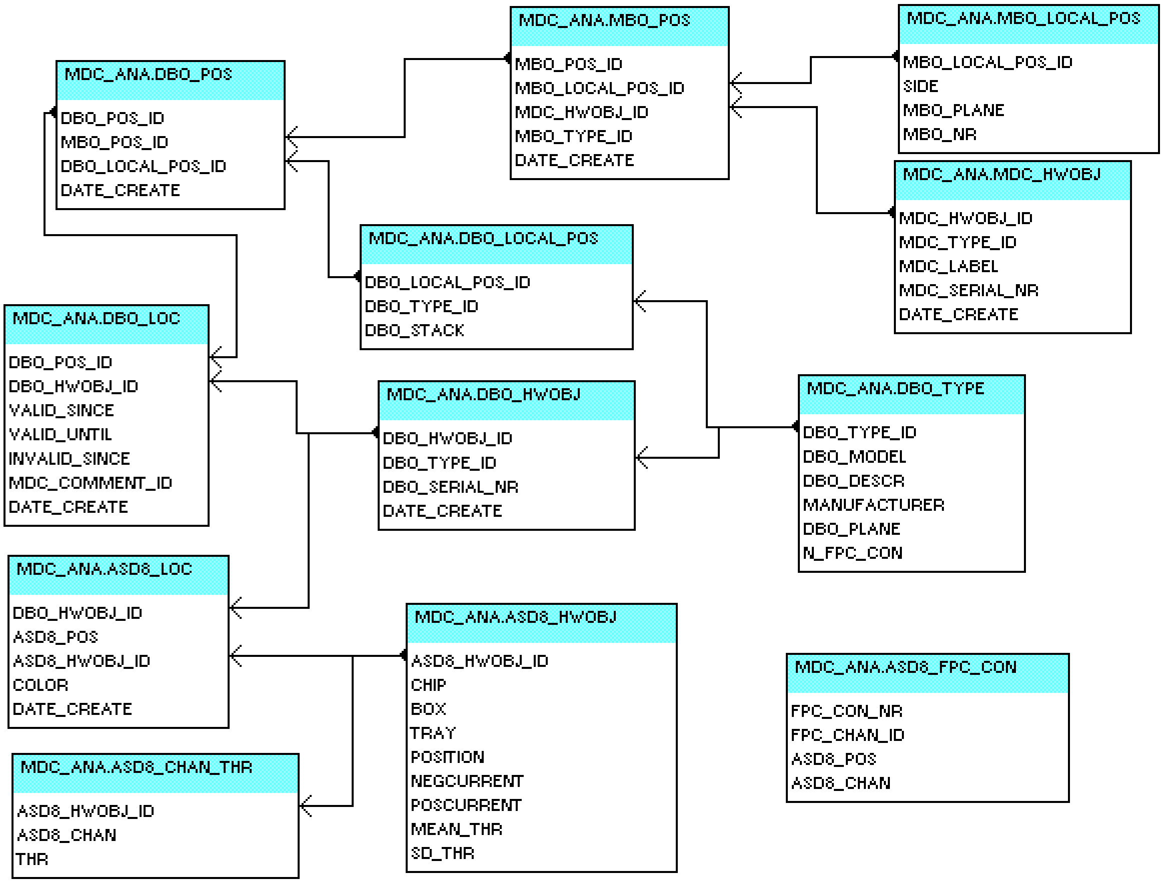
2.3 The daughterboards and the ASD8 chips

DBO_TYPE
Types of daughterboards
This table contains the description of the two daughterboard types, mounted at upper and lower position in the stack.

DBO_HWOBJ
List of all daughterboards (hardware)
This table contains at manufactured daughterboards.

DBO_LOCAL_POS
Positions of daughterboards on a motherboard (local scope)
This table contains the possible positions of the daughterboards in the stack, including the related type.

DBO_POS
Positions of daughterboards on all motherboards of the MDCs
This table stores all possible DBO positions on all possible MBO positions (independent from the MBO really mounted)



DBO_LOC

Placement of daughterboards on motherboard positions on the MDCs

The table contains the DBO hardware objects mounted.

A trigger firing on insert supports the version management and checks, that the entry is unique (only on object at the specified position, DBO not mounted at more than one position).

As for MBO_LOC a WebDB GUI allows to insert and change the last actual valid entries. Historic changes (before the last change of the table) must be done with SQL by an expert.

ASD8_HWOBJ

List of ASD8 chips (hardware)

This table contains all (3850) manufactured ASD8 chips with their original position in the delivery boxes, the mean currents and thresholds.

ASD8_CHAN_THR

Thresholds for channel 1 to 8 of ASD8 chips

This table contains also the measured thresholds for the individual channels.

ASD8_LOC

Position of ASD8 chips on daughtherboards

This table stores the position of 2820 ASD8 chips on the daughterboards, including the color code of the chip, characterizing the quality. This table has no version management.

ASD8_FPC_CON
Connection between ASD8 and FPC channel
This table stores the connection between the 16 FPC channels (4 connectors each with 4 channels) and the 16 ASD8 channels (2 chips left and right each with 8 channels) and a daughterboard.

Related views:

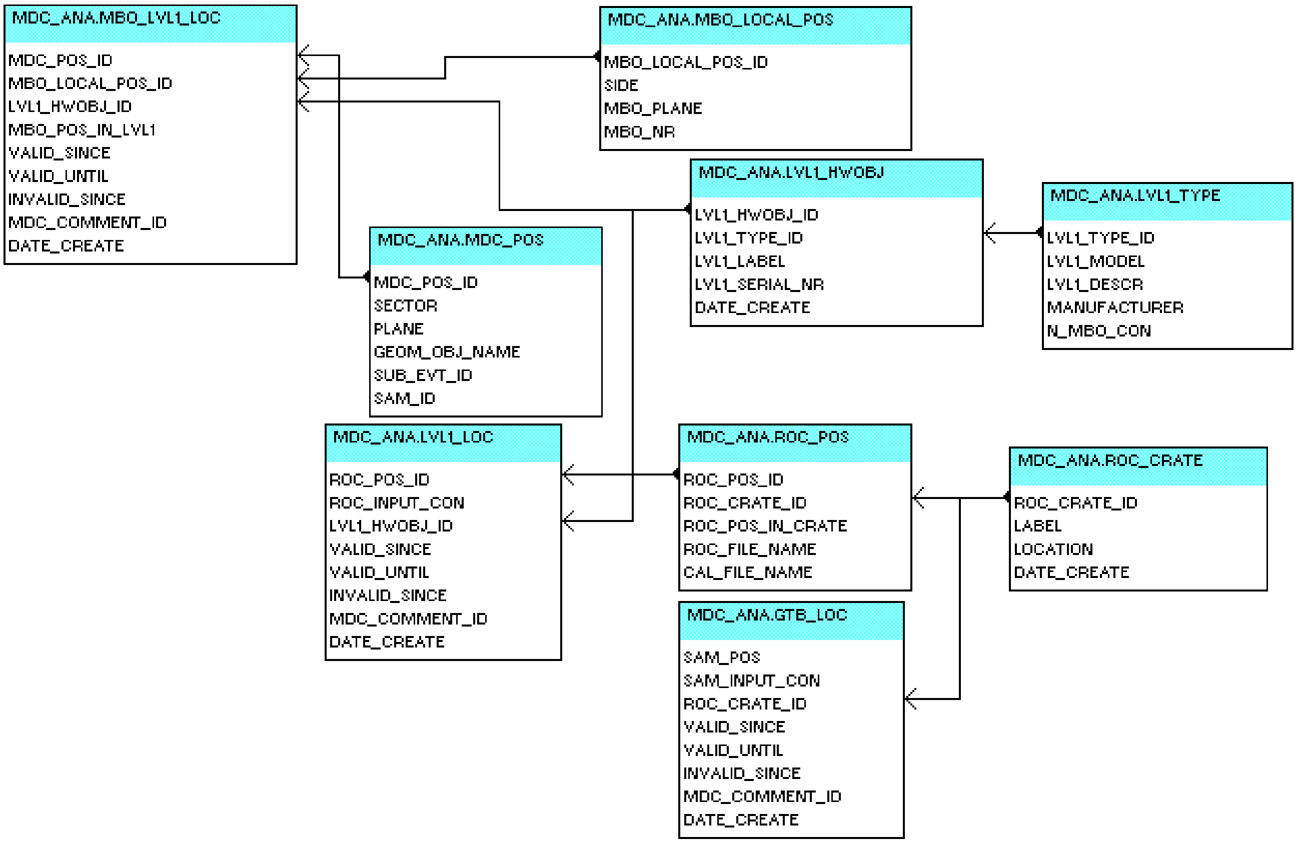
View name	*Description*
DBO_LOC_AT_DATE	All daughterboards mounted at a specified date
DBO_LOC_HISTORY	History of all mounted daughterboards
ASD8_LOC_ALL	Provides for both chips (one row per DBO) mounted on a daughterboard the original position in the storage tray
ASD8_THR_DATA	Provides for each ASD8 chip the thresholds of the individual channels, as well as the mean, range, minimum and maximum thresholds
ASD8_THR_PLACED	Combines the view ASD8_THR_DATA with the actual positions on the daughterboards
ASD8_THR_MOUNTED	Position and thresholds of ASD8 chips only mounted on daughterboards
WIRE_DB0_AT_DATE	Connects a wire to a TDC channel and an ASD8 chip, valid at a specified date
WIRE_TDC_DB0_CON_AT_DATE	Same as WIRE_DB0_AT_DATE

2.4 The readout hardware and cabling

LVL1_TYPE
Types of LVL1 Bus cables
This table contains actually only one entry. All columns are empty besides LVL1_type_id (the primary key) and N_MBO_CON (number of MBO connectors: 3).

LVL1_HWOBJ
LVL1 Bus cables
This table contains 260 cables with serial numbers.

MBO_LVL1_LOC
MBO-LVL1 cabling
This table stores the position of the LVL1 Bus cables on the (logical) MBO position, described by the MDC module position and the local MBO position on the module.
Although this table has a version management, the topology of this cabling hardly changes (different versions in the past were in fact wrong entries). Once filled properly, it will most probably not change and therefore needs no GUI for maintenance.



LVL1_LOC

LVL1-ROC cabling

This tables contains the other side of the LVL1 bus cables connected to a ROC, described by the logical ROC position and the input connector.
Because the cable may be connected to a different ROC port during a beam time, a WebDB GUI was developed to change the actual valid cabling via simple forms.

ROC_CRATE

Readout controller crates

The primary key roc_crate_id is identical with the MDC module position. The labels are not filled. Therefore this tables is used as a “position” table, not as a “hardware object” table. The lookup table MDC_POS_TO_ROC_CRATE_ID is therefore obsolete.

ROC_POS

Possible positions for the readout controllers

This table contains the possible positions of the readout controllers in the crate. The primary key ROC_POS_ID id coded as ROC_CRATE_ID (= MDC module position) * 10 +
ROC_POS_IN_CRATE (1..7).
The columns ROC_FILE_NAME, CAL_FILE_NAME are left-overs from B. Sailer's run-control.

GTB_LOC

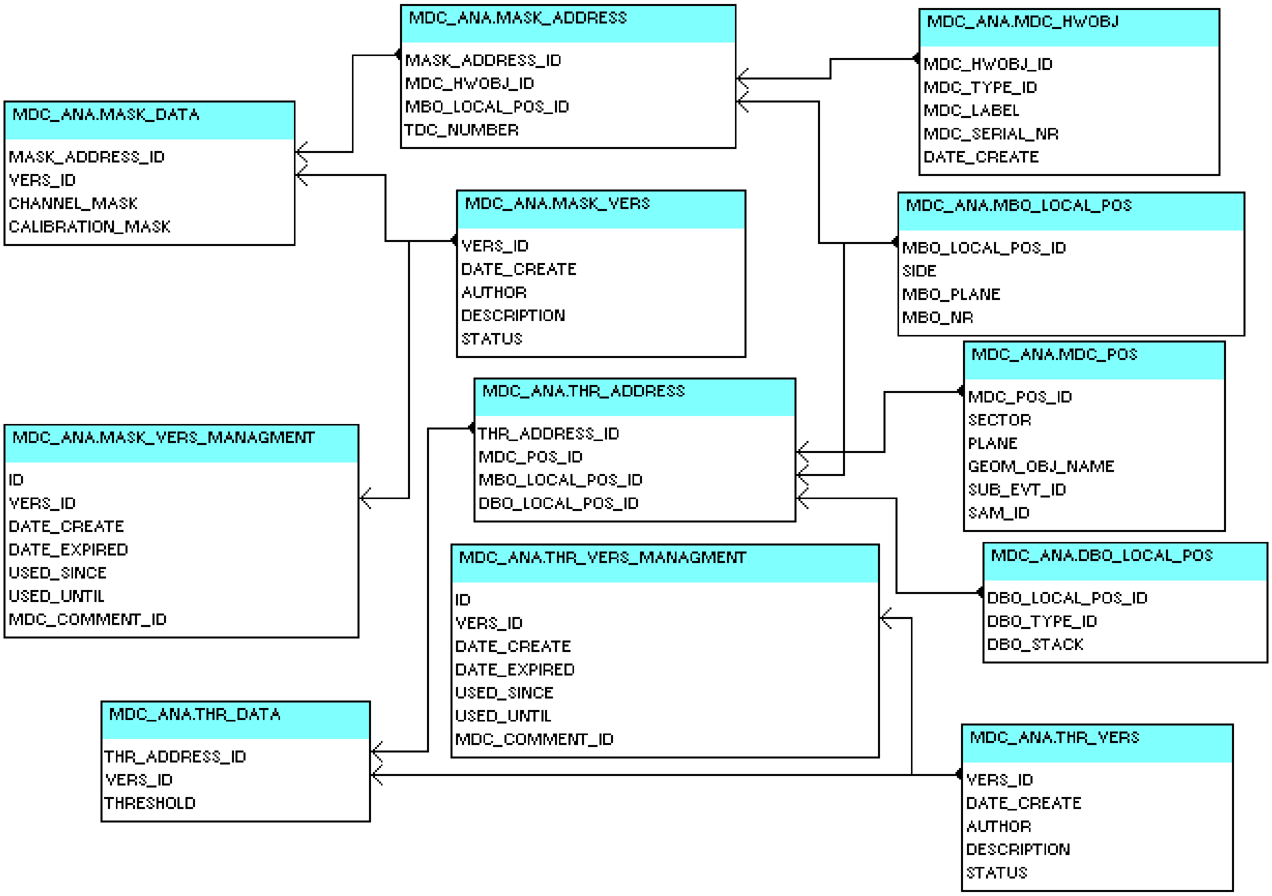
SAM-ROC connection

This tables contains the connection of the readout controllers on the SAM modules.
Necessary changes must be done by an expert, because a WebDB GUI was never developed.

Related views:

View name	*Description*
LVL1_LOC_ACTUAL	Actual valid connection MBO - ROC
MDCREADOUT_AT_RUN	Lookup table for the MDC unpacker in HYDRA (old version, no support of history date)
MDCREADOUT_AT_RUN_HIST	New version of the lookup table for the MDC unpacker in HYDRA
MDC_SUBEVENT_IDS_AT_DATE	Current subevent ids calculated from readout addresses
WIRE_ROC_ACTUAL	Lookup table wire – readout address

2.5 The readout parameters



Actually two groups of tables for the readout parameters exist: one for the TDC channels masks and one for the daughterboard thresholds. Four tables exist for each group:

1. a table defining the addresses of the data: xxx_ADDRESS

Address table	*Description*
MASK_ADDRESS	Logical address of TDC channel The addresses are coupled to the hardware MDC modules. Actually the table contains the addresses of all 26 chambers.
THR_ADDRESS	Logical positions of daugtherboards used for thresholds The addresses are coupled to the logical positions of the MDC modules (sector, plane).

2. a table defining the version of the data: xxx_VERS

   The status flag marks "usable" and "not usable" versions.

Version table	*Description*
MASK_VERSION	List of versions for the TDC channels masks
THR_VERSION	List of versions for the DBO Thresholds

3. a table holding the data for the different versions: xxx_DATA

Data table	*Description*
MASK_DATA	The data for the TDC channels masks
THR_DATA	The data for the DBO Thresholds

4. a table defining which version is valid/used at a certain date: xxx_VERS_MANAGMENT

Version management table	*Description*
MASK_VERS_MANAGMENT	Valid/used versions for the TDC channels masks
THR_VERS_MANAGMENT	Valid/used versions for the DBO Thresholds

There are two ways to store and change the data in Oracle, both using the same Oracle interface in HYDRA, library ora:

1. One may use the QT-GUI developed by Simon Lang, to read a version already existing in Oracle, to modify it, to write it back to Oracle (creates a new version) and to validate this version. This is preferred option!
2. One may also read a version valid for a specified run from Oracle with a ROOT macro and write it to an ASCII file, which then can be modified with an editor.
   With an other Macro one then reads the ASCII file and stores the data in Oracle (new version) and (eventually) validates it.

2.6 The analysis parameters

Besides the lookup tables, all analysis parameters are ether implemented as condition-style or tree-style. While the condition-style parameters are stored in tables in the account HANAL, the central analysis account, the tree-style parameters require the implementation of three tables for each parameter container:

1. a table ***xxx_VERS***, defining the version, author, comment, ...
2. a table ***xxx_DATA***
3. a table ***xxx***, defining the time range for which this version is/was valid.

While the first two tables are filled via analysis macros, the third table is filled via a generic WebDB interface, which also provides functionality to query the data including the history. Special views are required, all starting with "HWPG_". The other views (ending with "_AT_DATE") mentioned below are used by the current analysis interface.

(see also documentation http://www-hades.gsi.de/persons/ilse/ora_standard_container.html).

TDC calibration (CAL1):

Tables: CAL_METHOD, CAL1PAR_VERS, CAL1PAR_DATA, CAL1PAR

Views: HWPG_MDC_CAL1PAR_PARTS, HWPG_MDC_CAL1PAR_DATA,
MDC_CAL1PARVERS_AT_DATE

Drift time calibration (CAL2):

Tables: CAL2PARSIM_VERS, CAL2PARSIM_DATA, CAL2PARSIM

Views: HWPG_MDC_CAL2PARSIM_PARTS, HWPG_MDC_CAL2PARSIM_DATA,
MDC_CAL2PARSIM_VERS_AT_DATE

Cell efficiency:

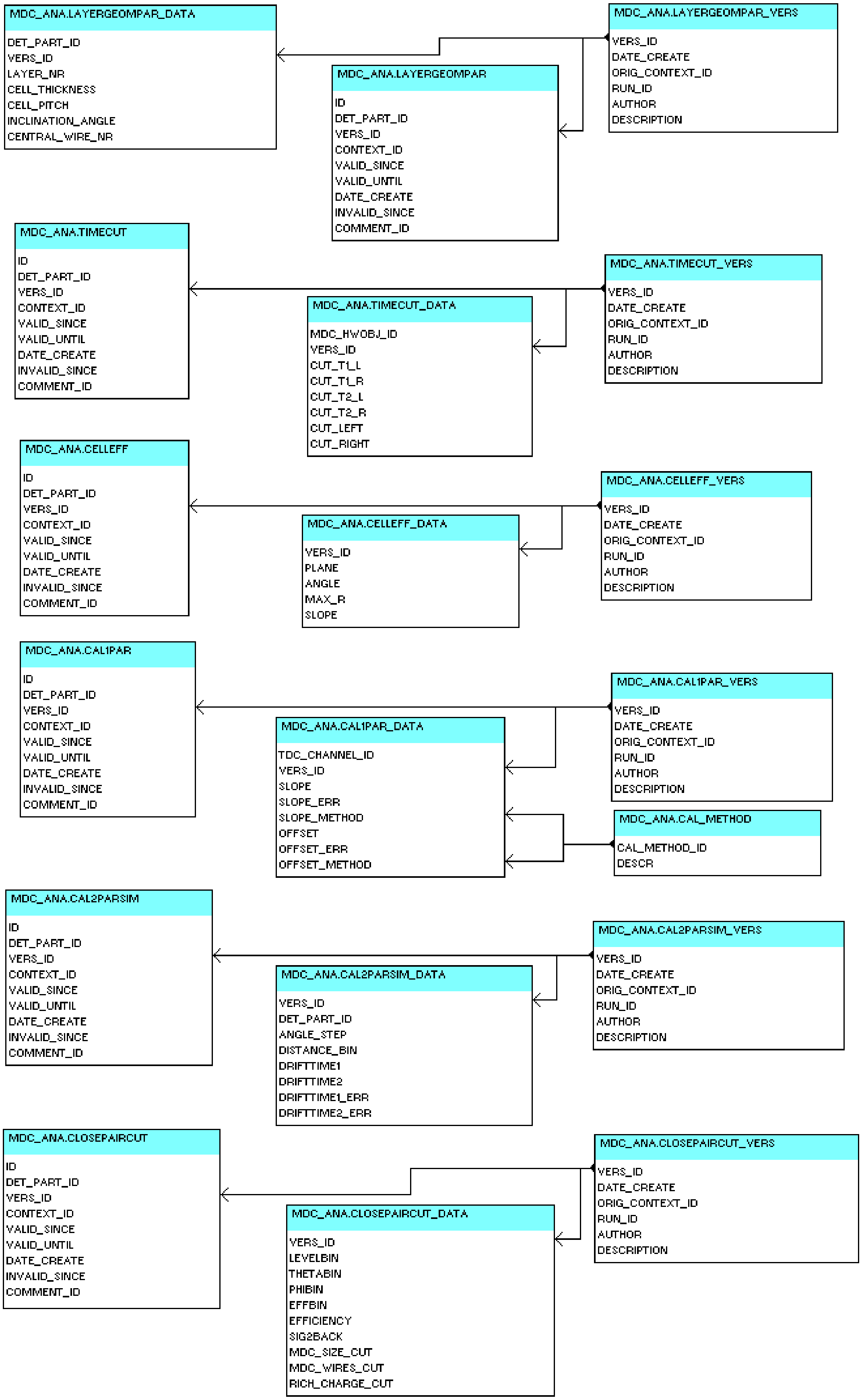
Tables: CELLEFF_VERS, CELLEFF_DATA, CELLEFF

Views: HWPG_MDC_CELLEFF_PARTS_DATA, MDC_CELLEFF_VERS_AT_DATE

Layer geometry:

Tables: LAYERGEOMPAR_VERS, LAYERGEOMPAR_DATA, LAYERGEOMPAR

Views: HWPG_MDC_LAYERGEOMPAR_PARTS, HWPG_MDC_LAYERGEOMPAR_DATA,
MDCLAYERGEOMPAR_AT_DATE


MDC_ANA.LAYERGEOMPAR_DATA
DET_PART_ID
VERS_ID
LAYER_NR
CELL_THICKNESS
CELL_PITCH
INCLINATION_ANGLE
CENTRAL_WIRE_NR
MDC_ANA.LAYERGEOMPAR_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.LAYERGEOMPAR
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.TIMECUT
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.TIMECUT_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.TIMECUT_DATA
MDC_HWOBJ_ID
VERS_ID
CUT_T1_L
CUT_T1_R
CUT_T2_L
CUT_T2_R
CUT_LEFT
CUT_RIGHT
MDC_ANA.CELLEFF
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.CELLEFF_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.CELLEFF_DATA
VERS_ID
PLANE
ANGLE
MAX_R
SLOPE
MDC_ANA.CAL1PAR
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.CAL1PAR_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.CAL1PAR_DATA
TDC_CHANNEL_ID
VERS_ID
SLOPE
SLOPE_ERR
SLOPE_METHOD
OFFSET
OFFSET_ERR
OFFSET_METHOD
MDC_ANA.CAL_METHOD
CAL_METHOD_ID
DESCR
MDC_ANA.CAL2PARSIM
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.CAL2PARSIM_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.CAL2PARSIM_DATA
VERS_ID
DET_PART_ID
ANGLE_STEP
DISTANCE_BIN
DRIFTTIME1
DRIFTTIME2
DRIFTTIME1_ERR
DRIFTTIME2_ERR
MDC_ANA.CLOSEPAIRCUT
ID
DET_PART_ID
VERS_ID
CONTEXT_ID
VALID_SINCE
VALID_UNTIL
DATE_CREATE
INVALID_SINCE
COMMENT_ID
MDC_ANA.CLOSEPAIRCUT_VERS
VERS_ID
DATE_CREATE
ORIG_CONTEXT_ID
RUN_ID
AUTHOR
DESCRIPTION
MDC_ANA.CLOSEPAIRCUT_DATA
VERS_ID
LEVELBIN
THETABIN
PHIBIN
EFFBIN
EFFICIENCY
SIG2BACK
MDC_SIZE_CUT
MDC_WIRES_CUT
RICH_CHARGE_CUT

Time cuts for noise reduction:

Tables: TIMECUT_VERS, TIMECUT_DATA, TIMECUT

Views: HWPG_MDC_TIMECUT_DATA

Close pair cuts:

Tables: CLOSEPAIRCUT_VERS, CLOSEPAIRCUT_DATA, CLOSEPAIRCUT

Views: HWPG_MDC_CLOSEPAIRCUT_DATA

This parameter container is not used anymore in the current analysis.

2.7 Tables used in the WebDB GUIs

Table	*Package*	*Comment*
COMP_HARDWARE_VERS	HARDWARE_MTN	GUI to change the MBO and DBO placement
COMP_MBO_LOC	MBO_MTN	GUI to change the MBO placement
COMP_DBO_LOC	DBO_MTN	GUI to change the DBO placement
COMP_READOUT_VERS	READOUT_MTN	GUI to change the readout cabling
COMP_LVL1_LOC	LVL1_LOC_MTN	GUI to change the LVL1 - ROC cabling

2.8 Obsolete tables

Table	*Comment*
LAYER_TYPE	Old table for layer geometry (without version management) used until July 2003

2.9 Packages

Packages used by triggers:

Package	*Description*
VERSMGM	Contains generic procedures to ensure the version management and is used by triggers
DBO_LOC_HANDLING	Originally developed to ease the version management of table DBO_LOC (placement of DBOs) and partially still used by trigger on insert
DBO_LOC_QUERY	Contains functions, which return the DBO object by address or the current address of an object
FPC_LOC_HANDLING	Originally developed to ease the version management of table FPC_LOC (placement of FPC cables) and partially still used by trigger on insert
FPC_LOC_QUERY	Contains functions, which return the FPC object by address or the current address of an object
LVL1_LOC_HANDLING	Originally developed to ease the version management of table LV1_LOC (placement of LV1 bus cables) and partially still used by trigger on insert
MBO_LOC_HANDLING	Originally developed to ease the version management of table MBO_LOC (placement of MBOs) and partially still used by trigger on insert
MBO_LOC_QUERY	Contains functions, which return the MBO object by address or the current address of an object
MBO_LVL1_LOC_HANDLING	Originally developed to ease the version management of table MBO_LVL1_LOC (cabling of MBO – LV1 bus) and partially still used by trigger on insert

Packages used WebDB GUI:

Package	*Description*
DBO_MTN	WebDB GUI to change DBOs
LVL1_LOC_MTN	WebDB GUI to change the cabling LV1 bus - ROCs
MBO_MNT	WebDB GUI to change MBOs
HARDWARE_MTN	WebDB GUI (menu and generic part) to change MBOs and DBOs
READOUT_MTN	WebDB GUI (menu and generic part) to change readout tables (actually only LV1 bus – ROCs)
HPS_PARAM_QUERY	WebDB GUI to show and compare the readout and lookup tables

Package	***Description***
	(integration in WebDB application “HYDRA Parameter Containers”)
MDC_UTIL	Utility package for WebDB GUI
MDC_ANA_DOC	WebDB documentation of the production account MDC_ANA
MDC_SETUP_INFO	WebDB GUI to show the MDC setup
MDC_SETUP_QUERY	WebDB GUI to show the detector setup, the mounting and cablings of the MBOs, DBOs, FPCs and the lookup table for the readout
THR_MASK_MTN	Maintenance for thresholds and channel masks (WebDB GUI and Run Control interface)
THR_MASK_COMP	Used by WebDB GUI to compare different threshold and channel mask versions
ASD8_INFO	WebDB GUI to show the data and placement of the ASD8 chips, the connection between the ASD8 channels and the wires, and the relations to the thresholds

Packages used by the analysis interface:

Package	***Description***
MDC_PAR_QUERY	Public interface for the analysis to create a new parameter version and to store thresholds and channel masks

3 The account MDC_WWW

Almost all applications made with the WebDB GUI designer have been replaced by self-written code.

The only package not replaced is CHART_ASD8_MEAN_VAL, which shows the chart of ASD8 mean thresholds. It will be discarded when WebDB is not supported any longer.