MainBoss CMMS Database Structure—Version 2.9, Update 9

(Permission granted to reproduce all or part of this document provided that Thinkage's copyright notice is included in the reproduction. No money may be charged for such reproductions except to recover all or part of the cost of reproduction.)

This document describes the database files used by MainBoss CMMS version 2.9, Update 9. See the section *Changes in MainBoss 2.9* for recent changes.

Important: This document is intended to help you read information from MainBoss database files. Under no circumstances should you ever change the information in any MainBoss file, except by using MainBoss itself.

The following general principles apply to all database files:

- Most files are stored directly under the MainBoss data directory. However, there are still the following subdirectories:
  - ARCLH contains compressed archive files
  - ARC contains de-archived database files
  - TEMP contains various temporary files created during regular operation
  - @Requests contains database files for the MainBoss @Requests program

- Information in the database can take a variety of forms:
  - Integers (numbers without fractional parts)
  - Floating point numbers, typically used for monetary values (although some monetary values are stored as an integer number of pennies)
  - Date values
  - Text (fixed-length character strings)
  - Memos (variable-length character strings)
  - Yes/No values, typically used for options that can either be on or off. The values are either `'T'` for true or `'F'` for false. (Some programs accept other values for yes/no data, such as `'0'` and `'1'` or `'Y'` and `'N'`, but MainBoss only uses `'T'` and `'F'`.)
  - Times are usually represented as text strings containing six digits, representing `HHMMSS`

- Database files are in Microsoft FoxPro 2.0 format.

- The first record (Physical Record 1) in certain files is a special "prototype" record. This record is reserved for use by MainBoss software. In a prototype record, the Key field gives the key number that should be assigned to the next new record created in the file. For example, in the WorkRequests file, the prototype record contains the identification number that should be assigned to the next work request created. When a new work request is created, it is given the key number from the prototype record, and the key in the prototype record is updated to the number that should be used in the next work request.

Prototype records may also supply defaults for some fields and sequence numbers for the user-visible work order, work request, and purchase order numbers.

You should never change the prototype record in an existing database file. Also, if you are creating a new table that requires a prototype record, the easiest method is to assign the record number of each new record to the key (starting at 1,000,000 plus the `RECNO` value). After that, change the key field of the prototype record to one more than the highest record number.

- Some fields are described as *hidden*. These are used during the process of upgrading from an old database format to the new one, and in certain operations that repair the existing database. The hidden field is used in records which appear to refer to other tables where the appropriate matching record does not exist. The repair process creates the missing record (or as much of the record as can be guessed), then marks the created record as hidden. The hidden record will not appear in regular browser lists and is allowed to have the same code as other records in the list.

This document uses a simple format to describe the data fields in each database file. Here is a typical entry:

```
 2 PRONAME    VPCODE     ProgramName  The program name (user assigned) [Text(50)]
```

This entry consists of the following:

- The number of the data field in the database file (in this case, field 2)
- The column name of the field (`PRONAME`)
- The name used when exporting the field in connection with a report (`VPCODE`)
- A more descriptive name for the field used in MainBoss programming code (`ProgramName`)
- A brief explanation of what the field contains
- The type of data in the field (in this case `TEXT`); for Text and Integer types, the entry gives the length of the field (so that `Text(50)` means a string of 50 characters).

Some data fields are "joins"—links to another database file. For example, the `UNITKEY` field in a Work Request record is a join to the Units table, pointing to the record for the unit where work is requested. If work is requested for a particular piece of equipment, `UNITKEY` joins to the Unit table entry for that equipment. We use the notation `J(Units->Key)` to indicate a field that is a join to the `Key` field in the Units database.

IMPORTANT: The database formats described in this document only apply to MainBoss 2.9. Future versions of MainBoss will make no effort to remain backward compatible with these formats. Therefore, if you write your own applications to read data directly from a MainBoss database file, you will have to update those applications with each new revision of MainBoss.

The end of this document contains an index that will make it easier to find the information you want.

Exported Data

When MainBoss exports data from a report (either in database format or text format), the fields in the exported data have names beginning with a prefix. The prefix tells what kind of record contained the field; the rest of the name is the name of that field within the database table. For example, suppose that exported data contains information from the Work Order table. To indicate that this data comes from this table, MainBoss uses a prefix of `WO`. Therefore the exported data has field names like `WONUM` (the `NUM` field in a work order record), `WOTYPE` (the `TYPE` field in a work order record), and so on.

The following prefixes are used in exported data:

Prefix	Table
AC	Access codes
AR	Archive sets
AS	Asset accounts
BU	Buildings
CC	Cost centers
CD	Closing codes
EX	Purchase order extras
IA	Adjustment codes
IC	Inventory categories
IS	Issue codes
IV	Inventory items
LA	Labor records (from work orders)
LO	Inventory locations
MH	Meter history
MT	Meters
OB	Obsolete codes
OW	Ownerships
PJ	Projects
PL	Personnel
PM	Permissions
PO	Purchase orders
PR	Priority codes
RC	Receipts
SC	Preventive maintenance schedules
SH	Shipping modes
SP	Spare parts
ST	Storerooms
SV	Service contracts
SY	System
TM	Payment terms
TP	Templates
TR	Trades
TS	Template specifications

TX	Inventory transactions
UC	Unit categories
UM	Units of measurement
UN	Units
VC	Vendor categories
VN	Vendor name
VP	View programs
WC	Work categories
WO	Work orders (including chargebacks)
WR	Work requests

In some cases, export data will contain two records of the same type. For example, the task report includes two types of tasks: "general" tasks (listed in the Tasks table) and "assigned unit" tasks (which are specific to a particular unit). Since both types of tasks are represented by work order records, fields in a general task have the prefix WO while fields in an assigned unit task have the prefix TWO. Similarly, in some reports there are multiple building records (for example, the building containing a unit and buildings containing spare parts for that unit); again, the first building in an export record will have the prefix BU while the next has the prefix TBU.

Changes in MainBoss 2.9

Table formats have not changed between this update and the previous one.

For MainBoss 2.9, Update 5, a new permission (ADMIN21) was implemented, allowing a user to export the database contents as XML.

A number of fields have been lengthened to allow for longer data values.

A new COMMENT field has been added to inventory item records in invitems.dbf.

This document describes tables used by MainBoss @Requests as well as those used by MainBoss CMMS.

Labor [contains prototype record]

The prototype record gives default values.

labor.dbf: Actual Personnel Labor records for Workorders

```
 1 WOKEY      -          WorkOrderKey J(WorkOrders->Key) [Integer(10)]
 2 LABORKEY   -          LaborKey     J(Personnel->Key) [Integer(10)]
 3 TRADE      LATRADE    Trade        J(Trades->Code) [Text(10)]
 4 STARTDATE  LADATE     StartDate    Date work started [Date]
 5 STARTTIME  LATIME     StartTime    HHMMSS time work started [Text(6)]
 6 TIME       LADUR      Time         HHHHMM time duration [Text(6)]
 7 LABORRATE  LARATE     LaborRate    Hourly labor rate in pennies [Integer(6)]
 8 ESTIMATE   LAESTFLG   Estimate     T if this is an estimated record [Logical]
```

EditPrograms [contains prototype record]

programs.dbf: Programs used to work with documents or drawings

1	KEY	VPKEY	Key	Unique key [Integer(10)]
2	PRONAME	VPCODE	ProgramName	The program name (user assigned) [Text(50)]
3	VIEWCMD	VPVIEWCMD	ViewCmd	The command line to View a document [Variable length text]
4	PRINTCMD	VPPRNTCMD	PrintCmd	The command line to Print a document [Variable length text]
5	STARTUP	VPSTARTUP	StartupFlag	Flag indicating the windowing to request [Text(1)]

Meters [contains prototype record]

meters.dbf: Meters used in scheduling assigned to a specific unit

1	KEY	MTKEY	Key	Unique key [Integer(10)]
2	CODE	MTCODE	Code	The meter code (user assigned) [Text(10)]
3	DESC	MTDESC	Desc	The meter description [Text(30)]
4	TYPE	MTTYPE	Type	The meter counting type: "C" means cumulative, "D" means direct measurement [Text(1)]
5	UNITKEY	-	UnitKey	J(Units->Key): which unit is this meter associated with [Integer(10)]
6	OFFSET	MTOFFSET	MeterReadingOffset	Meter reading offset applied to compute effective reading [Integer(12)]
7	UOMKEY	MTUOM	UoMKey	J(UoM->Key) Unit of measurement for meter [Integer(10)]
8	CREATEDATE	MTDATE	CreateDate	Create date [Date]
9	CREATETIME	MTTIME	CreateTime	Create time as HHMMSS [Text(6)]

MeterHistory

meterhst.dbf: Meter reading history

1	METERKEY	-	MeterKey	J(Meters->Key) Meter key associated with this record [Integer(10)]
2	DATE	MHDATE	Date	Meter record date [Date]
3	TIME	MHTIME	Time	Meter record time HHMMSS [Text(6)]
4	ORIGINKEY	-	OriginKey	J(WorkOrders->Key) for Work order entries; otherwise, the field is blank [Integer(10)]
5	READING	MHBREAD	Reading	Meter reading as read [Integer(12)]
6	EFFECTIVE	MHEREAD	EffectiveReading	Effective Meter reading including offset at time [Integer(12)]

Units [contains prototype record]

The Vendor fields identifying the Service Vendor will have a prefix of `'S'` (`SVNCODE`).

The Vendor fields associated with service contracts that are related to the unit are prefixed with `'C'` (`CVNCODE`).

The Vendor fields associated with the Purchase Vendor have no prefix.

units.dbf: Equipment and Space table

1	KEY	UNKEY	Key	Unique internal record key [Integer(10)]
2	UTYPE	UNTYPE	UnitType	"E" for Equipment or "S" for Space [Text(1)]
3	BUILDING	-	Building	J(Buildings->Key) [Integer(10)]
4	CODE	UNCODE	Code	Unit Code (User assigned) [Text(30)]
5	DESC	UNDESC	Desc	Unit Description [Text(50)]
6	LOCATION	UNLOC	Location	Current Unit location [Text(40)]
7	CATEGORY	UNCAT	UnitCategory	J(UnitType+UnitCategories->Code) [Text(10)]
8	MAKE	UNMAKE	Make	Unit brand/make [Text(30)]
9	MODEL	UNMODEL	Model	Model number [Text(30)]
10	SERIAL	UNSERIAL	Serial	Serial number [Text(30)]
11	DRAW	UNDRAW	Drawing	Drawing number [Text(10)]
12	PURCHASE	UNPURDATE	PurchaseDate	Purchase date [Date]
13	ORGCOST	UNORGCOST	OriginalCost	Original cost in pennies [Integer(12)]
14	REPCOST	UNREPCOST	ReplacementCost	Replacement cost in pennies [Integer(12)]
15	ESTIMATE	UNESTDATE	EstReplCostLastDate	Most recent date that the life expectancy and replacement cost were estimated [Date]
16	TLIFE	UNTLIFE	TypicalLife	Typical life span in years [Integer(3)]
17	VENDOR	-	PurchaseVendor	J(Vendors->Key): original vendor [Integer(10)]
18	SERVICE	-	ServiceVendor	J(Vendors->Key): service contractor [Integer(10)]
19	WEXPIRY	UNWARDATE	WExpiryDate	Warranty expiry date [Date]
20	OWNER	UNOWNER	Ownership	J(Ownerships->Code): Actual owner if external [Text(10)]
21	ASSET	UNASSET	AssetCode	J(AssetCodes->Code): Asset account for ownership [Text(15)]
22	COSTCENTER	UNCC	CostCenter	J(CostCenters->Code): Cost center for maintenance [Text(15)]
23	SYSTEM	UNSYSTEM	System	J(Systems->Code): System which uses Unit [Text(10)]
24	OBSOLETE	UNOBS	Obsolete	J(Obsoletes->Code): Obsolete code if obsolete [Text(10)]
25	DEPRATE	UNDEPRATE	DepreciationRate	Annual depreciation rate percentage for replacement schedule calculation [Integer(3)]
26	SCRAPD	UNSCPDATE	ScrapDate	Date of disposal (proposed or actual) [Date]
27	SCRAPV	UNSCPCOST	ScrapValue	Scrap Value (expected or actual) in pennies [Integer(12)]
28	CONTACT	NX	Contact	J(Contacts->Key) Person who operates/normally requests the work [Integer(10)]
29	ACCESS	UNACCESS	AccessCode	J(AccessCodes->Code) access restrictions on unit [Text(10)]
30	COMMENT	UNCOMMENT	Comment	Comment associated with Unit [Variable length text]
31	HIDDEN	-	Hidden	true if unit is hidden for use [Logical]

WorkRequests [contains prototype record]

wrequest.dbf: Work requests - each record represents a work request

1	KEY	WRKEY	Key	Internal work request number [Integer(10)]
2	WRNUM	WRNUM	Number	User visible work request number [Text(15)]
3	STATUS	WRSTATUS	Status	Status of the work request: blank means active, "T" means transferred, and "V" means void [Text(1)]
4	DATE	WRDATE	RequestDate	Request date [Date]
5	TIME	WRTIME	RequestTime	Request time HHMMSS [Text(6)]
6	REQUESTOR	-	Requestor	J(Contacts->Key) of requestor [Integer(10)]
7	UNITKEY	-	UnitKey	J(Units->Key) [Integer(10)]
8	WTYPE	WRWTYPE	WorkCategory	J(WorkCategories->Code) [Text(10)]
9	SUBJECT	WRSUBJECT	Subject	One line subject field for request [Text(60)]
10	ACCESSCODE	WRACCESS	AccessCode	J(AccessCodes->Code) [Text(10)]
11	PRIORITY	WRPRIOR	Priority	J(Priorities->Code) [Text(10)]

```
12 WDESC       WRDESC      Description  Text work description [Variable length text]
 13CLOSEDATE   WRCLSDATE   ClosingDate  Date request was closed [Date]
14 CLOSETIME   WRCLSTIME   ClosingTime  Time request was voided or transferred HHMMSS
                                        [Text(6)]
15 COMMENT     WRCOMMENT   Comment      Comment to requestor on disposition of request
                                        [Variable length text]
```

WorkOrders [contains prototype record]

This table holds work orders as well as task and "scheduled unit" information.

The `CLASS` and `TYPE` fields contain single characters indicating the nature of the work order. For normal work orders, the `TYPE` field may contain `O` (for open work orders), `C` (for closed work orders), or `V` (for voided work orders); otherwise it contains `T` for a task or `S` for a scheduled unit. The value stored in `LINKKEY` depends on `CLASS` and `TYPE`, as shown in the table below:

Work order type	TYPE	CLASS	LINKKEY
Corrective work order	`O,C,V`	`C`	blank or `J(WorkRequests->Key)`
Preventive work order (auto-generated)	`O,C,V`	`P`	`J(PreventiveSchedules->Key)`
Preventive work order (manually generated)	`O,C,V`	`M`	`J(WorkOrders->Key) Schedule`
Task work order entry	`T`	blank	blank
Scheduled work order	`S`	blank	`J(WorkOrders->Key) Task entry`

To find all workorders for a given request, use the `ByClassAndLink` tag with a `"C"+STR(WorkRequests->Key,10,0)` expression. To find all schedules associated with a particular Task entry, use the `ByTypeAndLink` tag with a `"S"+STR(WorkOrders->Key,10,0)` [where `WorkOrders->Key` refers to a `"T"` record].

worders.dbf: Work Orders - each record represents a work order

```
 1 KEY         WOKEY       Key          Internal work order number [Integer(10)]
 2 WONUM       WONUM       Number       User visible work order number [Text(15)]
 3 TYPE        WOTYPE      Type         See above table [Text(1)]
 4 CLASS       WOCLASS     Class        See above table [Text(1)]
 5 LINKKEY     WOLINK      LinkKey      See above table [Integer(10)]
 6 UNITKEY     -           UnitKey      J(Units->Key) [Integer(10)]
 7 WTYPE       WOWTYPE     WorkCategory J(WorkCategories->Code) [Text(10)]
 8 COSTCENTER  WOCC        CostCenter   J(CostCenters->Code) [Text(15)]
 9 PROJECT     WOPROJECT   Project      J(Projects->Code) [Text(10)]
10 ACCESSCODE  WOACCESS    AccessCode   J(AccessCodes->Code) [Text(10)]
11 PRIORITY    WOPRIOR     Priority     J(Priorities->Code) [Text(10)]
12 ORIGINATOR  -           Originator   J(Permissions->Key) of MainBoss user who created
                                        this work order [Integer(10)]
13 REQUESTOR   -           Requestor    J(Contacts->Key) of requestor [Integer(10)]
14 SUBJECT     WOSUBJECT   Subject      One line subject field for work order [Text(60)]
15 WDESC       WODESC      Description  Text work description [Variable length text]
16 CREATEDATE  WODATE      CreateDate   Create date [Date]
17 CREATETIME  WOTIME      CreateTime   Create time as HHMMSS [Text(6)]
18 CLOSEDATE   WOCLSDATE   ClosingDate  Closing date [Date]
19 CLOSETIME   WOCLSTIME   ClosingTime  Closing time as HHMMSS [Text(6)]
20 CLOSECODE   WOCLOSECD   ClosingCode  Closing code J(ClosingCodes->Code) [Text(10)]
21 CLOSECOMM   WOCOMMENT   ClosingComment Text of comment [Variable length text]
22 DOWNTIME    WODWNTIME   Downtime     Down time HHHHMM [Text(6)]
23 NEEDPRINT   WONEEDPRT   NeedPrint    Select for Print (True means yes) [Logical]
```

```
24 WORKDATE   WOWRKDATE  WorkStartDateWork start date [Date]
25 WINTERVAL  WOWRKDUR   WorkInterval Work interval (days) from WorkStartDate
                                      [Integer(6)]
```

ChargeBacks

chrgback.dbf: Chargeback information from closed work orders

```
 1 WOKEY      -          WorkOrderKey J(WorkOrders->Key) [Integer(10)]
 2 CHARGETO   WOCHRGTO   ChargeTo     Person to charge to [Text(30)]
 3 LABOR      WOLABCOST  LaborCost    Labor cost [Integer(10)]
 4 MATERIAL   WOMATCOST  MaterialCost Total Material cost [Integer(10)]
```

PreventiveSchedules [contains prototype record]

`RescheduleAlgorithm` contains one of the following digits:

```
 1 SingleFromClose       only one WO allowed at a time, reschedule from WO close
 2 SingleFromAnyClose    only one WO allowed at a time, reschedule from WO close for any
                                      WO on SRec
 3 SingleFromStart       only one WO allowed at a time, reschedule from WO start
 4 SingleFromAnyStart    only one WO allowed at a time, resched from WO start for *any*
                                      WO on SRec
 5 SingleFromSchedule    only one WO allowed at a time, resched from schedule point
```

The Schedule point is (`LASTSCHED + INTERVAL`).

WO close is `WORKDATE+WINTERVAL` or (estimated) meter reading on this date.

WO start is `WORKDATE` or meter (estimated) reading on this date.

schedule.dbf: Preventive maintenance schedules

```
 1 KEY        SCKEY      Key          Unique key for specific schedule [Integer(10)]
 2 WOKEY      SCTASK     WorkOrderKey J(WorkOrders->Key) link to srec (task instance for
                                      unit) [Integer(10)]
 3 TYPE       SCTYPE     ScheduleType "D" means days interval, "M" means month interval,
                                      "R" for meter reading [Text(1)]
 4 METERKEY   SCMETER    MeterKey     J(Meters->Key) if meter schedule; otherwise blank
                                      [Integer(10)]
 5 LASTSCHED  SCLAST     LastScheduled  Meter Reading if Metered; otherwise, date as
                                      integer (e.g. 19990727) [Integer(12)]
 6 INTERVAL   SCINTRVAL  ScheduleInterval     Scheduling interval in meter units or days
                                      for date scheduling [Integer(10)]
 7 SEASONBEG  SCSEABEG   SeasonStart  blank or MMDD of first day of on season in year
                                      [Text(4)]
 8 SEASONEND  SCSEAEND   SeasonEnd    blank or MMDD of last day of on season in year
                                      [Text(4)]
 9 INHSEASON  SCSEAINH   SeasonInhibit  True means inhibit schedule in off season, False
                                      means defer if off season [Logical]
10 WEEKMASK   SCWEEKMSK  WeekMask     Lower 4 bits of each byte represent days of week:
                                      bit 0 not used; Sunday assumed as first day;
                                      Wednesday-Saturday in byte 1, Sunday-Tuesday in
                                      byte 2 mask for week schedule with base character
                                      of 0x40 added. Note that all days on would show up
                                      as ASCII word ON [Text(2)]
```

```
11 INHWEEK    SCWEEKINH  WeekInhibit  True means inhibit schedule in week mask, False
                                      means defer if in week mask [Logical]
12 ALGORITHM  SCALGTHM   RescheduleAlgorithm  See table above [Text(1)]
13 INHIBIT    SCINH      InhibitSchedule      True means never schedule this entry,
                                      False means normal scheduling [Logical]
14 LASTWOMADE -          LastGeneratedWOKey   J(WorkOrders->Key) link to most-recently-
                                      generated work order [Integer(10)]
15 LATEINHIB  SCLATEINH  InhibitIfOverdue     True means inhibit if current date past
                                      schedule date, False means defer to current date
                                      [Logical]
```

InvTransactions [contains prototype record]

The first record in the table is a prototype record, used to hold the seed `Key` which links TT/TF and PC/EV record pairs.

Record types are defined by the `TYPE` field, which must be one of:

`PC`	Physical Count linked to associated EV by equal Reason fields.
`EV`	Evaluation (replaces total value) [no storeroom]
`TF`	Transfer From linked to matching TT by equal Reason fields
`TT`	Transfer To linked to matching TF by equal Reason fields
`IS`	Issued
`RC`	Received under Receipt(Reason)
`AJ`	Adjusted
`PO`	Purchase under P/O(Reason) [storeroom only advisory]
`RS`	Reserved for W/O(Reason) [storeroom only advisory]
`RT`	Reserved for TASK W/O(Reason) [storeroom only advisory]
`WO`	Issued to W/O(Reason)
`DQ`	MainBoss-generated SR quantity correction from upgrader.
`DV`	MainBoss-generated overall value correction from upgrader [no storeroom].
`PQ`	Price quote from a vendor (Reason is vendor code), like PO records, quantity and total cost are negative.

When the type is `PO`, the Quantity is negative and indicates the amount ordered. Storeroom, location are not currently used (but may become a recommendation to the receiver as to where to send the goods when they arrive). Since the Quantity is negative, the `PO` and `RC` records for a single purchase order can be totaled straight to determine remaining quantity on order. In most cases, `PO` records do not participate in totaling operations, as their quantities do not represent actual inventory changes. The `Reason` field contains the Key of a Purchases record.

When the type is `PC`, the Quantity indicates a new total on-hand in the given location, and the `TotalCost` field contains the previous assumed quantity. If the `PC` involved a total value adjustment using an `EV` record, these are linked to each other by matching `Reason` fields.

When the type is EV, the TotalCost indicates the new total value, and the Quantity contains the previous assumed quantity. If the EV is a result of a PC, it is linked to the PC by matching Reason fields.

For other record types, both fields are deltas, giving the change in the associated quantity.

TT and TF records should appear in pairs side-by-side with equal and opposite Quantities. They are linked by equal Reason fields.

When the type is IS, the Reason field contains the reason for the issue (taken from issues.dbf). The Location field contains the person issued to (just as text).

When the type is AJ, the Reason field contains a user-defined adjustment code (from adjusts.dbf).

When the type is WO or RS or RT, the Reason field contains the work order key ('T' record for RT).

When the type is RC, the Reason field contains the Receipt record key number.

When the Type is PQ, the Reason field contains the Vendor code.

invtx.dbf: Inventory transactions - each record is one change to an inventory item

1	ITEMKEY	TXITEM	InvItem	J(InvItems->Key): Inventory item affected [Integer(10)]
2	TYPE	TXTYPE	Type	Transaction type [Text(2)]
3	DATE	TXDATE	Date	Date of transaction [Date]
4	TIME	TXTIME	Time	Time of transaction HHMMSS [Text(6)]
5	QTY	TXQTY	Quantity	Amount in quantity field meaning depends on TYPE [Integer(11)]
6	TOTALCOST	TXCOST	TotalCost	Total signed cost in cents [Integer(11)]
7	REASON	TXREASON	Reason	J(various tables) [Text(10)]
8	STRMKEY	TXSTRM	Storeroom	J(InvLocations->Storeroom) and J(Storerooms->Key) [Integer(10)]
9	LOC	TXLOC	Location	e.g. bin number [Text(20)]
10	REFER	TXREFER	Reference	Reference number for transaction [Text(30)]

InvLocations

Inventory on hand by location. These represent a summation of the entries in InvTx.dbf.

invlocs.dbf: Inventory on hand by location - each record is one storeroom/item combination

1	STRMKEY	LOSTRM	Storeroom	J(Storerooms->Key) [Integer(10)]
2	ITEMKEY	LOITEM	InvItem	J(InvItems->Key) [Integer(10)]
3	LOC	LOLOC	Location	Location within storeroom [Text(20)]
4	ONHAND	LOONHAND	OnHandQty	Total on hand for this location [Integer(10)]

Purchase [contains prototype record]

This file contains the vendor, account, and scheduling information required to group together multiple PO inventory transactions and possibly multiple receipt records (each holding RC transactions). The PO can be associated with a work order using type X extra cost records, in which case in which case type L and M Extra Cost records are allowed.

This file contains a prototype record, and a unique key field. The status of the purchase order depends on the single-character value in the STATE field:

Purchase order type	STATE
Draft	D
Completed (finished)	F
Issued to vendor	I
Closed normally	C
Prototype purchase order	P

Voided purchase orders are flagged when the VOID field is set to True.

purchase.dbf: Purchase Order table

```
 1 KEY         POKEY      Key           Unique internal purchase order key [Integer(10)]
 2 PONUM       PONUM      Number        User visible Purchase Order Number [Text(15)]
 3 VENDOR      -          Vendor        J(Vendors->Key) vendor [Integer(10)]
 4 ORDERNUM    POORDNUM   VendorOrderNumber    Vendor's order number [Text(40)]
 5 COSTCENTER  POCC       CostCenter    J(CostCenters->Code) cost center to charge to
                                        [Text(15)]
 6 NEEDED      PONEDDATE  DateRequired  Date goods are required [Date]
 7 EXPECTED    POEXPDATE  DateExpected  Date vendor expects to ship goods [Date]
 8 COMMENT     POCOMMENT  Comment       Comments [Variable length text]
 9 STATE       POSTATE    State         State of this purchase order [Text(1)]
10 VOID        POVOID     Void          True if PO voided, otherwise False [Logical]
11 CREATED     PODATE     XCreateDate   Date Order created/req submitted [Date]
12 CREATET     POTIME     XCreateTime   Time Order created/req submitted [Text(6)]
13 COMPLETED   POCOMDATE   CompleteDate  Date order complete (ready to send out) [Date]
14 COMPLETET   POCOMTIME   CompleteTime  Time order complete (ready to send out) [Text(6)]
15 SUBMITD     POSUBDATE   SubmitDate    Date order sent to vendor [Date]
16 SUBMITT     POSUBTIME   SubmitTime    Time order sent to vendor [Text(6)]
17 CLOSED      POCLSDATE   CloseDate     Date order closed (complete or not) [Date]
18 CLOSET      POCLSTIME   CloseTime     Time order closed (complete or not) [Text(6)]
19 SHIPPING    POSHIP      Shipping      J(ShippingModes->Code): shipping mode [Text(10)]
20 TERMS       POTERMS     Terms         J(Terms->Code): payment terms [Text(10)]
21 VCOMMENT    POVCOMM     VendorComment Comment to vendor on printed PO [Variable length
                                         text]
22 SHIPTO      POSHIPTO    ShipTo        J(Buildings->Key): shipping address key for PO
                                         [Integer(10)]
23 DESC        PODESC      Summary       Short description of PO [Text(60)]
24 NEEDPRINT   PONEEDPRT   NeedPrint     Select for Print [Logical]
25 ORIGINATOR  -           Originator    J(Permissions->Key) of MainBoss user who created
                                         this purchase order [Integer(10)]
```

Receipts [contains prototype record]

This file contains the record of a single incoming shipment. It records the date received, along with shipping information, and links to the corresponding PO record. The items received are entered as RC transactions. The receiver should at least fill in the quantities. Other information such as the storeroom can be filled in later by a storeroom keeper.

This file contains a prototype record, and a unique key field.

receive.dbf: Shipments Received table

1	KEY	RCKEY	Key	Unique internal Receipt key [Integer(10)]
2	POKEY	RCPOKEY	POKey	J(Purchases->Key) link to PO record [Integer(10)]
3	DATE	RCDATE	Date	Date shipment received [Date]
4	TIME	RCTIME	Time	Time shipment received HHMMSS [Text(6)]
5	WAYBILL	RCWAYBILL	Waybill	Shippers shipment number or waybill [Text(15)]
6	EXTRACOST	RCEXCOST	ExtraCost	Extra cost/discount for this shipment beyond sum of items [Integer(10)]
7	COMMENT	RCCOMMENT	Comment	Comment entered at time of receiving [Variable length text]

Payment Terms

Contains the payment terms mentioned in purchase orders. The user selects by CODE, and the CODE appears in the printed PO Payment Terms box.

terms.dbf: Payment terms table

1	CODE	TMCODE	Code	Code [Text(10)]
2	DESC	TMDESC	Desc	Description [Text(30)]

ShippingModes

Contains the shipping modes mentioned in Purchase Orders. The user selects by CODE and the DESC appears in the printed PO Shipping Instructions box.

shipping.dbf: Shipping modes table

1	CODE	SHCODE	Code	Code [Text(10)]
2	DESC	SHDESC	Desc	Description [Text(40)]

PurchaseExtras [contains prototype record]

Contains slave records off purchase orders for extra line items that are not inventory items and therefore may not have any quantity. QTY and UOMKEY can be both blank or both non-blank. If they are non-blank and QTY is non-zero, a cost-per-item can be calculated.

Only estimate records have unique KEY values generated from the prototype record.

Actual records are indicated by setting ESTIMATE to False. These contain the key values of the associated estimate records. KEY will be blank if the Actual record is not associated with any estimate records.

poextras.dbf: Purchase Extra line items table

1	KEY	EXKEY	Key	Unique identifier for this item [Integer(10)]
2	POKEY	EXPOKEY	POKey	J(Purchases->Key) link to PO record [Integer(10)]
3	QTY	EXQTY	Quantity	Quantity purchased [Integer(10)]
4	UOMKEY	EXUOM	UoMKey	J(UoM->Key) Unit of Measure [Integer(10)]
5	COST	EXCOST	TotalCost	Total cost for item [Integer(10)]
6	DESC	EXDESC	Desc	Item description [Text(40)]
7	CATNO	EXCATNO	CatalogNumber	Vendors catalogue number [Text(20)]
8	ESTIMATE	EXESTFLG	Estimate	T if this is an estimated cost [Logical]
9	TYPE	EXTYPE	Type	"U" for user entry, "M" for Work Order Materials entry, "L" for Work Order Labor entry, "X" for PO/WO linkage [Text(1)]
10	WOKEY	EXWOKEY	WorkOrder	J(WorkOrders->Key) corresponding WO for type "M" and "L" records [Integer(10)]

SpareParts

sparepts.dbf: Equipment spare parts

1	EQUIPKEY	SPUNITKEY	Unit	J(Units->Key) [Integer(10)]
2	ITEMKEY	SPITEMKEY	Item	J(InvItems->Key) [Integer(10)]
3	QTY	SPQTY	Quantity	Quantity of part to be reserved [Integer(10)]

Specifications

For pathname specifications, the `INFO` field will contain the pathname and the `TYPE` field will contain a `'P'`. `TKEY` in this case refers to the EditPrograms key entry for the viewing/printing program.

For templated specifications (based on a template from template.dbf), `TYPE` will contain `T` and the `INFO` memo field will contain the encoded field information. Each field encoding is

```
n:XXXXXXXX
```

where `n` is the length in base 10 ASCII (`"5"` means 5 characters) of the number of characters that follow the `":"` delimiter. The `XXXXX` is the actual value for the field in the specification as entered by the user.

The field values appear in the same order as the fields in the template in a left to right, top to bottom manner. A null field is represented with `"0:"`.

specs.dbf: Specifications for assets - one record for filled out information

1	ASSOCKEY	TSASSOC	AssociationKey	J(<table>->Key): Depends on which table is using specification [Integer(10)]
2	TYPE	TSTYPE	Type	Type of specification: "T" for templated spec, "P" for pathname association [Text(1)]
3	CODE	TSCODE	Code	User assigned code for identification [Text(10)]
4	TKEY	TSTEMPKEY	TemplateKey	J(Templates->Key) from when instantiated [Integer(10)]
5	INFO	TSINFO	Info	Specification additional information (Field info for template, pathname for pathname association) [Variable length text]

ServiceContracts [contains prototype record]

svctable.dbf: Service contracts

1	KEY	SVKEY	Key	Unique key [Integer(10)]
2	CODE	SVCODE	Code	User code for contract [Text(15)]
3	DESC	SVDESC	Desc	User descripton for contract [Text(30)]
4	VENDOR	-	Vendor	J(Vendors->Key) [Integer(10)]
5	STARTDATE	SVBEGDATE	StartDate	Beginning date for contract [Date]
6	EXPIRYDATE	SVENDDATE	EndDate	Ending date for contract [Date]
7	PART	SVPARTS	PartsFlag	True if Parts covered, False otherwise [Logical]
8	LABOR	SVLABOR	LaborFlag	True if Labor covered, False otherwise [Logical]
9	COST	SVCOST	Cost	Service cost in pennies [Integer(11)]
10	COMMENT	SVCOMMENT	Comments	Comments on service contract [Variable length text]

ServicedEquipment

Equipment under service contract. Each record describes coverage for one piece of equipment with one service contract.

svcequip.dbf: Equipment under service contract

1	SVCKEY	-	ServiceContract	J(ServiceContracts->Key) [Integer(10)]
2	EQUIPKEY	-	Unit	J(Units->Key) [Integer(10)]

Buildings [contains prototype record]

building.dbf: Building table

1	KEY	BUKEY	Key	Internal unique Key [Integer(10)]
2	CODE	BUCODE	Code	Building Code [Text(15)]
3	DESC	BUDESC	Desc	Building Description [Text(50)]
4	ADDRESS1	BUADDR1	Address1	Building street address line 1 [Text(30)]
5	ADDRESS2	BUADDR2	Address2	Building street address line 2 [Text(30)]
6	CITY	BUCITY	City	Building city [Text(30)]
7	TERRITORY	BUPROV	Territory	Building state/province [Text(20)]
8	COUNTRY	BUCOUNTRY	Country	Building country [Text(20)]
9	PCODE	BUPOSTAL	PostalCode	Building postal/zip code [Text(10)]
10	HIDDEN	-	Hidden	True if no longer visible to users [Logical]

CostCenters

costctr.dbf: Cost center table

1	CODE	CCCODE	Code	Cost Center Code [Text(15)]
2	DESC	CCDESC	Desc	Cost Center Description [Text(40)]

AssetCodes

assets.dbf: Asset code table

1	CODE	ASCODE	Code	Asset Code Code [Text(15)]
2	DESC	ASDESC	Desc	Asset Code Description [Text(40)]

VendorCategories

vendcat.dbf: Vendor category table

```
 1 CODE       VCCODE     Code         Vendor category Code [Text(10)]
 2 DESC       VCDESC     Desc         Vendor category Description [Text(30)]
```

Vendors [contains prototype record]

vendors.dbf: Vendor table

```
 1 KEY        VNKEY      Key          Internal Unique Key [Integer(10)]
 2 CODE       VNCODE     Code         Vendor Code [Text(15)]
 3 DESC       VNNAME     Name         Vendor Name [Text(40)]
 4 CONTACT    VNCONTACT  Contact      Contact name of vendor [Text(30)]
 5 CATEGORY   VNCAT      Category     J(VendorCategories->Code) [Text(10)]
 6 ADD1       VNADDR1    Address1     User-defined additional info 1 [Text(30)]
 7 ADD2       VNADDR2    Address2     User-defined additional info 2 [Text(30)]
 8 ADD3       VNADDR3    Address3     User-defined additional info 3 [Text(30)]
 9 TEL        VNBUSFONE  BusPhone     Telephone number - voice [Text(42)]
10 FAX        VNFAXFONE  FaxPhone     Telephone number - Fax [Text(42)]
11 WEB        VNWEB      WebUrl       Vendor's Web address [Text(128)]
12 ACCOUNTNUM VNCUSTNUM  AccountNumber  Customer AccountNumber with Vendor [Text(30)]
13 EMAIL      VNEMAIL    Email        Email address of Vendor [Text(128)]
14 HIDDEN     -          Hidden       True if no longer visible to users [Logical]
15 COMMENT    VNCOMMENT  Comments     User Comments on Vendor [Variable length text]
```

AccessCodes

access.dbf: Access code table

```
 1 CODE       ACCODE     Code         Access Code Code [Text(10)]
 2 DESC       ACDESC     Desc         Access Code Description [Text(30)]
```

UnitCategories

Unit categories are expected to be unique within `UTYPE+CODE`.

unitcat.dbf: Unit category table

```
 1 UTYPE      UCTYPE     UnitType     "E" for equipment or "S" for Space [Text(1)]
 2 CODE       UCODE      Code         Unit Category Code [Text(10)]
 3 DESC       UCDESC     Desc         Unit Category Description [Text(30)]
```

Trades

trades.dbf: Trade table

```
 1 CODE       TRCODE     Code         Trade Code [Text(10)]
 2 DESC       TRDESC     Desc         Trade Description [Text(30)]
```

Personnel [contains prototype record]

person.dbf: Personnel table

```
 1 KEY         PLKEY      Key           Internal unique Key [Integer(10)]
 2 CODE        PLCODE     Code          Person Code [Text(15)]
 3 DESC        PLNAME     Name          Person Name [Text(40)]
 4 TRADE       PLTRADE    Trade         J(Trades->Code): Person's trade [Text(10)]
 5 BTEL        PLBUSFONE  BusPhone      Business telephone number [Text(42)]
 6 HTEL        PLHOMFONE  HomePhone     Home telephone number [Text(42)]
 7 PAGER       PLPAGFONE  PagerPhone    Pager number [Text(42)]
 8 MOBILE      PLMOBFONE  MobilePhone   Mobile number [Text(42)]
 9 CURRATE     PLCURRATE  CurrentRate   Current hourly rate in pennies [Integer(6)]
10 NEWRATE     PLNEWRATE  NewRate       New pay rate in pennies [Integer(6)]
 11EFFDATE     PLNEWDATE  EffectiveDateEffective date for new rate [Date]
12 HIDDEN      -          Hidden        True if no longer visible to users [Logical]
13 EMAIL       PLEMAIL    Email         Email address of Personnel [Text(128)]
14 COMMENT     PLCOMMENT  Comments      User Comments on Personnel [Variable length text]
```

Contacts [contains prototype record]

contacts.dbf: Contacts table

```
 1 KEY         CNKEY      Key           Internal unique Key [Integer(10)]
 2 CODE        CNCODE     Code          Contact Name [Text(50)]
 3 BTEL        CNBUSFONE  BusPhone      Business telephone number [Text(42)]
 4 HTEL        CNHOMFONE  HomePhone     Home telephone number [Text(42)]
 5 PAGER       CNPAGFONE  PagerPhone    Pager number [Text(42)]
 6 MOBILE      CNMOBFONE  MobilePhone   Mobile number [Text(42)]
 7 HIDDEN      -          Hidden        True if no longer visible to users [Logical]
 8 EMAIL       CNEMAIL    Email         Email address of Contact [Text(128)]
 9 COMMENT     CNCOMMENT  Comments      User Comments on Contact [Variable length text]
10 LANGID      CNLANGID   LanguageId    4 digit hexadecimal encoded LANGID of contact's
                                        preferred language [Text(4)]
```

Priorities

priority.dbf: Priority table

```
 1 CODE        PRCODE     Code          Priority Code [Text(10)]
 2 DESC        PRDESC     Desc          Priority Description [Text(30)]
```

WorkCategories

workcat.dbf: Work category table

```
 1 CODE        WCCODE     Code          Work Category Code [Text(10)]
 2 DESC        WCDESC     Desc          Work Category Description [Text(30)]
```

ClosingCodes

closings.dbf: Closing code table

```
 1 CODE        CDCODE     Code          Closing Code Code [Text(10)]
 2 DESC        CDDESC     Desc          Closing Code Description [Text(30)]
```

Systems

systems.dbf: System table

```
 1 CODE        SYCODE      Code          System Code [Text(10)]
 2 DESC        SYDESC      Desc          System Description [Text(30)]
```

Ownerships

owners.dbf: Ownership table

```
 1 CODE        OWCODE      Code          Ownership Code [Text(10)]
 2 DESC        OWDESC      Desc          Ownership Description [Text(30)]
```

Obsoletes

obsolete.dbf: Obsolete code table

```
 1 CODE        OBCODE      Code          Obsolete Code [Text(10)]
 2 DESC        OBDESC      Desc          Obsolete Description [Text(30)]
```

Templates [contains prototype record]

Templates are referenced by `KEY` field. Specifications always refer to their parent template with the `KEY` value. This means a template cannot be altered/deleted while it is in use by a specification. To allow changes to templates after they are in use, any modification to a template will create a "new" template (a different key value) with the same `CODE` and `DESC`. However, only the most recent template is "visible" for selection. The older template is marked not visible, and will be purged from the table when its last use is deleted.

Each template fields are contiguous sequences of underscore (`"_"`) characters in the Layout field. Each such underscore represents one character of the field. Lines are separated with line breaks.

template.dbf: Templates - each record represents one template

```
 1 KEY         TPKEY       Key           Template Key [Integer(10)]
 2 CODE        TPCODE      Code          Template Code [Text(10)]
 3 DESC        TPDESC      Desc          Template Description [Text(30)]
 4 HIDDEN      -           Hidden        True if template is hidden for use [Logical]
 5 PLATE       TPPLATE     Layout        Original Template Layout [Variable length text]
```

Storerooms [contains prototype record]

stores.dbf: Storeroom table

```
 1 KEY         STKEY       Key           Unique storeroom key [Integer(10)]
 2 BUILDING    STBUILDING  Building      J(Buildings->Key) Building where storeroom located
                                         [Integer(10)]
 3 CODE        STCODE      Code          Storeroom Code [Text(10)]
 4 DESC        STDESC      Desc          Storeroom Description [Text(30)]
 5 HIDDEN      -           Hidden        True if storeroom is hidden for use [Logical]
```

InvCategories

invcat.dbf: Inventory category table

```
 1 CODE        ICCODE      Code          Inventory Category Code [Text(10)]
 2 DESC        ICDESC      Desc          Inventory Category Description [Text(30)]
```

UoM [contains prototype record]

uom.dbf: Units of measurement table

```
 1 CODE        UMCODE      Code          UoM Code [Text(10)]
 2 DESC        UMDESC      Desc          UoM Description [Text(30)]
 3 KEY         UMKEY       Key           Unique internal record key [Integer(10)]
```

InvItems [contains prototype record]

All of the `LAST...` fields were taken out; they can all be found by a scan relation into the Inv TX file using the reverse-date tags.

The current inventory value is now a total value to avoid rounding errors. Both values are stored as pennies.

`TOTALSEQ` starts at zero for a new record, and is incremented once before a transaction is written for the item, and incremented again after all the totals have been validated, all under a record lock on the inventory item. As a result, finding an odd `TOTALSEQ` in a locked item record means a process was aborted while totaling, and the totals must be redone from scratch.

invitems.dbf: Inventory item table

```
 1 KEY         IVKEY       Key           Unique key [Integer(10)]
 2 CODE        IVCODE      Code          Item Code [Text(30)]
 3 DESC        IVDESC      Desc          Item Description [Text(50)]
 4 CAT         IVCAT       Category      J(InvCategories->Code) [Text(10)]
 5 UOMKEY      IVUOM       UoMKey        J(UoM->Key) Unit of measure [Integer(10)]
 6 MINI        IVMIN       MinStock      Minimum stock (order point) [Integer(10)]
 7 MAXI        IVMAX       MaxStock      Maximum stock (to determine recommended order)
                                         [Integer(10)]
 8 TOTALCOST   IVCOST      TotalCost     Total cost of in-stock amount in cents
                                         [Integer(12)]
 9 ONHAND      IVONHAND    OnHand        Quantity on hand [Integer(12)]
10 ONORDER     IVONORDER   OnOrder       Quantity on order in active POs [Integer(12)]
11 ONRESERVE   IVONRES     OnReserve     Quantity on reserve for open WorkOrders
                                         [Integer(12)]
12 TOTALSEQ    -           TotalSeq      Sequence number incremented when transaction
                                         changed [Integer(12)]
13 INVENTORY   IVINVFLAG   Inventoried   True if this is a inventoried item otherwise false
                                         [Logical]
14 COMMENT     IVCOMMENT   Comment       Comment associated with Item [Variable length
                                         text]
```

Issues

issues.dbf: Issue code table

```
 1 CODE       ISCODE     Code         Issue Code [Text(10)]
 2 DESC       ISDESC     Desc         Issue Description [Text(30)]
```

Adjustments

adjusts.dbf: Adjustment code table

```
 1 CODE       IACODE     Code         Adjustment Code [Text(10)]
 2 DESC       IADESC     Desc         Adjustment Description [Text(30)]
```

Projects

projects.dbf: Project table

```
 1 CODE       PJCODE     Code         Project Code [Text(10)]
 2 DESC       PJDESC     Desc         Project Description [Text(30)]
```

Permissions [contains prototype record]

users.dbf: MainBoss Users and permissions - each record defines the permission for a particular user

```
 1 KEY        PMKEY      Key          The user key [Integer(10)]
 2 NAME       PMNAME     UserName     The user name [Text(30)]
 3 PASSWORD   -          Password     The user's password [Text(10)]
 4 STATUS     PMSTATUS   Status       The user's status: "A" for active, "D" for
                                      Disabled, "H" for Hidden [Text(1)]
 5 L11        -          WorkRequestPermissionCM_WO_OPEN_REQUEST [Logical]
 6 L12        -          L12          CM_WO_OPEN_WORKORDER [Logical]
 7 L13        -          L13          CM_WO_BROWSE_REQUESTS [Logical]
 8 L14        -          L14          CM_WO_BROWSE_WORKORDERS [Logical]
 9 L15        -          L15          CM_INV_NEWPO [Logical]
10 L16        -          L16          CM_WO_PRINT [Logical]
11 L17        -          L17          CM_REP_MAINT_MANPOWER [Logical]
12 L18        -          L18          CM_REP_MAINT_WR_HISTORY [Logical]
13 L19        -          L19          CM_TABLE_INV_SHIPPING [Logical]
14 L21        -          L21          CM_MAINT_BROWSE_TASKS [Logical]
15 L22        -          L22          CM_INV_NEWRECEIPT [Logical]
16 L23        -          L23          0 [Logical]
17 L24        -          L24          0 [Logical]
18 L25        -          L25          CM_REP_UNITS_MAINTENANCE_STATUS [Logical]
19 L26        -          L26          CM_TABLE_CONTACTS [Logical]
20 L27        -          L27          CM_MAINT_GENERATE_WORKORDERS [Logical]
21 L28        -          L28          CM_REP_UNITS_SPAREPARTS [Logical]
22 L29        -          L29          CM_REP_INV_PURCHASE_HISTORY [Logical]
23 L31        -          L31          CM_EQUIP_EQUIPMENT [Logical]
24 L32        -          L32          CM_EQUIP_METERS [Logical]
25 L33        -          L33          CM_EQUIP_CONTRACTS [Logical]
26 L34        -          L34          0 [Logical]
27 L35        -          L35          CM_TABLE_INV_TERMS [Logical]
28 L36        -          L36          CM_EQUIP_OBSOLETE [Logical]
29 L37        -          L37          CM_TABLE_VIEWPROGRAMS [Logical]
30 L38        -          L38          CM_INV_ADJUSTMENTS_QUOTE [Logical]
31 L39        -          L39          0 [Logical]
```

```
32 L41        -          PurchasePermission   CM_INV_PURCHASING [Logical]
33 L42        -          L42          CM_INV_RECEIVING [Logical]
34 L43        -          L43          CM_INV_ADJUSTMENTS_ISSUE [Logical]
35 L44        -          L44          CM_INV_ADJUSTMENTS_COUNT [Logical]
36 L45        -          L45          0 [Logical]
37 L46        -          L46          0 [Logical]
38 L47        -          L47          CM_INV_ADJUSTMENTS_OTHER [Logical]
39 L48        -          L48          CM_INV_ADJUSTMENTS_TRANSFER [Logical]
40 L49        -          L49          CM_INV_ADJUSTMENTS_VALUE [Logical]
41 L410       -          L410         CM_REP_UNITS_DOWNTIME [Logical]
42 L51        -          L51          CM_REP_MAINT_OPEN_WO [Logical]
43 L52        -          L52          CM_REP_MAINT_WO_OVERDUE [Logical]
44 L53        -          L53          CM_REP_MAINT_WO_HISTORY [Logical]
45 L54        -          L54          CM_REP_MAINT_LABOR_HISTORY [Logical]
46 L55        -          L55          CM_REP_MAINT_PRINT_TASKS [Logical]
47 L56        -          L56          0 [Logical]
48 L57        -          L57          0 [Logical]
49 L58        -          L58          CM_REP_MAINT_CONTRACTED_SERVICE [Logical]
50 L59        -          L59          CM_REP_MAINT_CHARGE_BACKS [Logical]
51 L510       -          L510         CM_REP_MAINT_MAINT_JOURNAL [Logical]
52 L511       -          L511         CM_REP_MAINT_WO_STATISTICS [Logical]
53 L512       -          L512         0 [Logical]
54 L513       -          L513         0 [Logical]
55 L514       -          L514         0 [Logical]
56 L515       -          L515         0 [Logical]
57 L516       -          L516         CM_REP_UNITS [Logical]
58 L517       -          L517         0 [Logical]
59 L518       -          L518         0 [Logical]
60 L519       -          L519         CM_REP_UNITS_REPLACEMENT_SCHEDULE [Logical]
61 L520       -          L520         CM_REP_INV_RE_ORDER [Logical]
62 L521       -          L521         CM_REP_INV_ON_ORDER [Logical]
63 L522       -          L522         CM_REP_INV_RECEIVING [Logical]
64 L523       -          L523         CM_REP_INV_ISSUES [Logical]
65 L524       -          L524         CM_REP_INV_WHEREUSED [Logical]
66 L525       -          L525         CM_REP_INV_STOCK_ACTIVITY [Logical]
67 L526       -          L526         CM_REP_INV_INVENTORY_STATUS [Logical]
68 L527       -          Login        CM_SESSION_LOGIN [Logical]
69 L528       -          L528         CM_SESSION_SELECT [Logical]
70 L529       -          L529         CM_SESSION_SELECT_MAIN [Logical]
71 L530       -          L530         CM_REP_INV_INVENTORY_VALUE [Logical]
72 L61        -          L61          CM_TABLE_BUILDINGS [Logical]
73 L62        -          L62          CM_TABLE_COST_CENTERS [Logical]
74 L63        -          L63          CM_TABLE_ASSET_ACCOUNTS [Logical]
75 L64        -          L64          CM_TABLE_VENDOR_TYPES [Logical]
76 L65        -          L65          CM_TABLE_VENDORS [Logical]
77 L66        -          L66          CM_TABLE_MAINT_ACCESS [Logical]
78 L67        -          L67          CM_TABLE_MAINT_SPACE_TYPE [Logical]
79 L68        -          L68          CM_TABLE_MAINT_SPACES [Logical]
80 L69        -          L69          CM_TABLE_MAINT_TRADES [Logical]
81 L610       -          L610         CM_TABLE_MAINT_PERSONNEL [Logical]
82 L611       -          L611         CM_TABLE_MAINT_PRIORITIES [Logical]
83 L612       -          L612         CM_TABLE_MAINT_WORK_CATEGORIES [Logical]
84 L613       -          L613         CM_TABLE_MAINT_REPAIR_CLOSING_CODES [Logical]
85 L614       -          L614         CM_TABLE_EQUIP_EQUIPMENT_TYPES [Logical]
86 L615       -          L615         CM_TABLE_EQUIP_SYSTEMS [Logical]
87 L616       -          L616         CM_TABLE_EQUIP_OWNERSHIPS [Logical]
88 L617       -          L617         CM_TABLE_EQUIP_OBSOLETE_CODES [Logical]
89 L618       -          L618         CM_TABLE_EQUIP_TEMPLATE [Logical]
90 L619       -          L619         CM_TABLE_INV_STOREROOMS [Logical]
91 L620       -          L620         CM_TABLE_INV_CATEGORIES [Logical]
92 L621       -          L621         CM_TABLE_INV_UNITS_OF_MEASURE [Logical]
93 L622       -          L622         CM_TABLE_INV_ITEMS [Logical]
94 L623       -          L623         CM_TABLE_INV_ISSUES [Logical]
```

```
 95 L624       -          L624         CM_TABLE_INV_ADJUSTMENTS [Logical]
 96 L625       -          L625         CM_TABLE_PROJECTS [Logical]
 97 L626       -          L626         0 [Logical]
 98 ADMIN1     -          RegisterUser CM_REGISTER_USER [Logical]
 99 ADMIN2     -          Admin2       CM_PREF_WORKORDERS [Logical]
100 ADMIN3     -          Admin3       CM_PREF_PO [Logical]
101 ADMIN4     -          Admin4       CM_PREF_GENERAL [Logical]
102 ADMIN5     -          Admin5       CM_PREF_CUSTOMTEXT [Logical]
103 ADMIN6     -          Admin6       CM_SECURITY_CHANGEPASSWORD [Logical]
104 ADMIN7     -          Admin7       CM_SECURITY_USERS [Logical]
105 ADMIN8     -          Admin8       CM_ARCH_BROWSE [Logical]
106 ADMIN9     -          Admin9       CM_ARCH_BACKUP [Logical]
107 ADMIN10    -          Admin10      CM_ARCH_RESTORE [Logical]
108 ADMIN11    -          Admin11      CM_ARCH_CLEAR [Logical]
109 ADMIN12    -          ReindexPermission    CM_DATABASE_REINDEX [Logical]
110 ADMIN13    -          Admin13      CM_DATABASE_BACKUP [Logical]
111 ADMIN14    -          Admin14      CM_DATABASE_RESTORE [Logical]
112 ADMIN15    -          AuditPermission      CM_DATABASE_AUDIT [Logical]
113 ADMIN16    -          Admin16      CM_BROWSE_LICENSE_KEYS [Logical]
114 ADMIN17    -          Admin17      0 [Logical]
115 ADMIN18    -          Admin18      CM_PREF_UNITS [Logical]
116 ADMIN19    -          Admin19      CM_PREF_WORKREQUESTS [Logical]
117 ADMIN20    -          UpgradePermission    CM_DATABASE_UPGRADE [Logical]
118 ADMIN21    -          Admin21      0 CM_DATABASE_EXPORTXML [Logical]
119 ADMIN22    -          Admin22      0 [Logical]
120 ADMIN23    -          Admin23      0 [Logical]
121 ADMIN24    -          Admin24      0 [Logical]
122 ADMIN25    -          LastAdminPermission  0 [Logical]
123 L121       -          L121         0 [Logical]
124 L122       -          L122         0 [Logical]
125 L123       -          L123         0 [Logical]
126 L124       -          L124         0 [Logical]
127 L125       -          L125         0 [Logical]
128 L126       -          L126         0 [Logical]
129 L127       -          L127         0 [Logical]
130 L128       -          L128         0 [Logical]
131 L129       -          L129         0 [Logical]
132 L130       -          L130         0 [Logical]
133 L131       -          L131         0 [Logical]
134 L132       -          L132         0 [Logical]
135 L133       -          L133         0 [Logical]
136 L134       -          L134         0 [Logical]
137 L135       -          L135         0 [Logical]
138 L136       -          L136         0 [Logical]
139 L137       -          L137         0 [Logical]
140 L138       -          L138         0 [Logical]
141 L139       -          L139         0 [Logical]
142 L140       -          LastPermissionField  0 [Logical]
```

Defaults

default.dbf: System default values

```
 1 LGEND      -          LastPmGenerateDate   Last time preventive maintenance work
                         orders were generated [Date]
 2 GFREQ      -          PmGenerateFrequency  PM generation frequency [Text(3)]
 3 PHONEPREX  -          PhonePrefix  Phone number prefix [Text(16)]
 4 EDITPKEY   -          EditProgramKey J(EditPrograms->Key): Default document program
                         [Integer(10)]
 5 SERIAL     -          RegistrationNumber   Program serial number [Text(10)]
 6 COMPANYNAM -          CompanyName  Installing company (program owner) [Text(30)]
 7 COMPANYADD -          CompanyAddress Company Address [Text(250)]
```

```
 8 BILLADDR    -          BillingAddress Bill-to address in POs [Text(250)]
 9 POCONTACT   -          PurchaserContact     Purchaser contact info for PO [Text(250)]
10 REINDEX     -          ReindexPhase Current Reindex phase or 0 if completed
                                  [Integer(5)]
11 PRFORMAT    -          PRSequenceFormat     PR Sequence number format [Text(15)]
12 POFORMAT    -          POSequenceFormat     PO Sequence number format [Text(15)]
13 WRFORMAT    -          WRSequenceFormat     WR Sequence number format [Text(15)]
14 WOFORMAT    -          WOSequenceFormat     WO Sequence number format [Text(15)]
15 CURRSYM     -          CurrencySymbol Currency Symbol String [Text(10)]
16 FRDATE      -          FmtRptDate   Report date output format [Text(30)]
17 FRCPOS      -          FmtRptCurrPositive   Positive Currency Format string for
                                  reports [Text(20)]
18 FRCNEG      -          FmtRptCurrNegative   Negative Currency Format string for
                                  reports [Text(20)]
19 FRCDEC      -          FmtRptCurrDecimalCharacter  Character to separate units and
                                  tenths in currency in reports [Text(1)]
20 FRCGROUP    -          FmtRptCurrDigitGroupSeparator     Character to separate digit
                                  groups in currency in reports [Text(1)]
21 FRCGSIZE    -          FmtRptCurrDigitGroupSize    Number of digits in a digit group
                                  in currency in reports [Integer(1)]
22 FRCLGSIZE   -          FmtRptCurrDigitLastGroupSize      Number of digits in last
                                  digit group in currency in reports [Integer(1)]
23 FCDORDER    -          FmtControlDateOrder  Field order 0:YMD 1:MDY 2:DMY [Integer(1)]
24 FCDDELIM    -          FmtControlDateDelimiter     0:none 1:/ 2:blank or comma
                                  [Integer(1)]
25 FCDMFORM    -          FmtControlDateMonthFormat   0:2 digits 1:abbreviation
                                  2:fullname [Integer(1)]
26 FCTAMPM     -          FmtControlTimeAmPm   True for AM mode false for 24 hour
                                  [Logical]
27 DBVERSION   -          DBVersion    Fake version of MainBoss database tables to deter
                                  old MB programs [Text(3)]
28 MINVERSION  -          DBMinVersion Fake version of MainBoss database tables to deter
                                  old MB programs [Text(1)]
29 UPGRSTEP    -          DBUpgrStep   Fake version of MainBoss database tables to deter
                                  old MB programs [Text(1)]
```

Settings

This file is where MainBoss stores information when you save the settings for a dialog window (i.e. by right-clicking on the dialog and selecting **Save Settings**.

Settings.dbf: MainBoss dialog customized settings

```
 1 IDD         -          DialogName   Typically IDD_... [Text(32)]
 2 IDC         -          ControlName  Typically IDC_... [Text(32)]
 3 ORIGINATOR  -          Originator   J(Permissions->Key) of MainBoss user for whom this
                                  value is used blank indicates site/default use
                                  [Integer(10)]
 4 VALUE       -          SettingValue The string representation of the setting
                                  [Text(64)]
```

ArchivedSets [contains prototype record]

Each record represents the contents of one archive set.

archives.dbf: Archived sets

```
 1 KEY         ARKEY      Key          Unique internal work request key [Integer(10)]
```

2	CODE	ARCODE	Code	Archive code [Text(10)]
3	BACKDATE	ARDATE	CreationDate	Date archive was created [Date]
4	S_DATE	ARBEGDATE	StartDate	Start date of archived data [Date]
5	E_DATE	ARENDDATE	EndDate	End date of archived data [Date]
6	DESC	ARDESC	Desc	User-supplied comment for archive [Text(50)]
7	LZHFILE	ARLZHFILE	FileName	Name of compressed file [Text(15)]
8	DBVERSION	-	DBVersion	Version of MainBoss database tables [Text(23)]
9	RESTORED	ARRESDATE	RestoredDate	date when archive set was restored or blank if not [Date]
F	10	ARCHIVING	-	Archiving True if in process of being archived, otherwise False [Logical]

@Requests Accounts [contains prototype record]

This file is used by @Requests to list the email addresses of authorized requestors. All requestors have entries in the MainBoss Contacts table, so the record contains a link to the contact information.

@Requests\accounts.dbf: Accounts - each record represents a requestor account

1	KEY	KEY	Key	Requestor Key [Integer(10)]
2	FLAGS	FLAGS	Flags	Flags [Integer(10)]
3	CONTACTKEY	CKEY	ContactKey	J(Contacts) [Integer(10)]

@Requests Logs [contains prototype record]

This file is used by @Requests to record log entries.

@Requests\logs.dbf: Logs - each record represents an email (work) request log

1	KEY	KEY	Key	Internal log number [Integer(10)]
2	DETAILS	DETAILS	Details	Details associated with request [Variable length text]
3	MESSAGE	MESSAGE	Message	Original email message [Variable length text]
4	HISTORY	HISTORY	History	History of Status/Error [Variable length text]
5	STATUS	STATUS	Status	Status value [Integer(10)]
6	OLDSTATUS	OLDSTATUS	OldStatus	Old status value [Integer(10)]
7	WOSCHEDULE	WOSCHELD	WoSchedule	Work Order schedule for the request [Text(15)]
8	WRKEY	KEY	WRKey	J(WorkRequests) [Integer(10)]
9	FLAGS	FLAGS	Flags	Flags [Integer(10)]
10	CDATE	CDATE	CrDate	Creation mail date [Date]
11	CTIME	CTIME	CrTime	Creation time in HHMMSS format [Text(6)]

Index

W