# Generic vLink Database Structure

## Login Security

The database login procedure is handled using the standard ASP.NET membership tables that will handle and encrypt logins and passwords.

aspnet\_Membership – stores user passwords, emails, reset hints, etc..  
aspnet\_User – stores the user name, and id

These tables are managed by the database and should not be adjusted by any procedure or SQL statement. The only field that is transferred is the UserName field in the spnet\_User table. The UserName field is linked to the username field in the VLinkUser table.

Although the database membership system also have user roles that can be defined, this is not currently used by the VLink applicatons.

## Database VLink Tables

|  |  |  |
| --- | --- | --- |
| **Table: VLinkUnit** | | |
| UnitID | Int32 | Unique identifier for each Vlink unit |
| Name | varchar(50) | Name associated with this Vlink unit |
| Active | Bit | Whether the unit is active |
| CurrentAlarm | Int32 | Reference to a currently active alarm in the alarm table (nullable) |
| SerialNumber | varchar(50) | Serial number of the VLink unit, used by VL\_TCP to get unit id number for VLink |

|  |  |  |
| --- | --- | --- |
| **Table: VLinkUser** | | |
| UnitID | Int32 | Vlink unit (from VLinkUnit table) |
| UserName | varchar(50) | Username associated with this unit, linked to UserName in aspnet\_User table |

Note: This table will store the relationships between the users and the V-Link units: a many-to-many relationship.

|  |  |  |
| --- | --- | --- |
| **Table: VLinkProperties** | | |
| UnitID | Int32 | Vlink unit (from VLinkUnit table) |
| ChangeDate | DateTime | Time when property is set |
| Property | Int32 | Property id number |
| Value | Varchar(50) | Property Value |

|  |  |  |
| --- | --- | --- |
| **Table: VLinkChannels** | | |
| UnitID | Int32 | Vlink unit (from VLinkUnit table) |
| Channel | Int32 | Channel number (1-30) |
| SensorID | Int32 | VMC Sensor ID Number |
| BusType | Int32 | Bus Type (J1708,CAN,….) |
| PGN | Int32 | PGN or PID or … |
| Address | Int32 | Source address (J1939) |
| Start | Int32 | Start location of data |
| Size | Int32 | Size of data |

|  |  |  |
| --- | --- | --- |
| **Table: VLinkActions** | | |
| id | Int32 | Unique identifier for each action |
| UnitID | Int32 | VLink unit to perform the action |
| Source | Varchar(50) | Who requested the action (nullable) |
| ActionType | Int16 | Enumeration for the type of action |
| Setting | Varbinary(max) | Data for the required action |
| RequestTime | DateTime | Time the action was posted |
| Pending | DateTime | Time when VL-TCP has action |
| CompleteTime | DateTime | Time when action completed |

Note: This table stored actions to be completed by an active VLink unit.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **Table: VLinkDataConversion** | | | | id | Int64 | Unique identifier for each record | | UnitID | Int32 | VLink unit identifier | | SensorID | Int32 | VMC Sensor ID number | | Conversion | Int16 | Enumerator of conversion types | | Value1 | Real | Double value (nullable) | | Value2 | Real | Double value (nullable) |   **Table: VLinkDataPackets** | | |
| id | Int64 | Unique identifier for each packet |
| UnitID | Int32 | VLink unit identifier |
| PacketDate | Datetime | Date that the packet was sent |
| DataType | Int16 | Enumerator of data types, VMC Sensor ID numbers |
| DataValue1 | Int32 | Simple 32-bit value (nullable) |
| DataValue2 | Real | Double value (nullable) |
| DataString | Varchar(255) | String data (nullable) |

|  |  |  |
| --- | --- | --- |
| **Table: VLinkAlarms** | | |
| id | Int32 | Unique identifier for each alarm/fault |
| UnitID | Int32 | VLink unit identifier |
| AlarmID | Int32 | Enumerator for alarm/fault type |
| BeginDate | Datetime | time that the alarm/fault was sent |
| EndDate | Datetime | time that the alarm/fault ended |
| AlarmString | Varchar(255) | String data (nullable) |
| AcknowledgeDate | DateTime | Time when the alarm was acknowledged by end user |
| AcknowledgeBy | varchar(50) | User that acknowledged the alarm |
| AlarmValue | Real | Alarm value (nullable) |

|  |  |  |
| --- | --- | --- |
| **Table: VLinkAlarmConfig** | | |
| id | Int32 | Unique identifier for each alarm/fault |
| UnitID | Int32 | VLink unit identifier |
| AlarmID | Int32 | Enumerator for the alarm/fault type |
| SensorID | Int32 | Enumerator for the sensor that faulted, client defined alarms |
| LowLimit | Real | Low Limit for client defined alarms |
| HighLimit | Real | High Limit for client defined alarms |
| OnAction | Varchar(255) | Action to take when alarm occurs |
| OffAction | Varchar(255) | Action to take when the alarm stops |
| Text | Varchar(255) | Text to include with email/text message |
| LinkedAlarm | Int32 | Alarm that must also be active for this alarm to trigger |

# Stored Procedures in the database

## proc\_connect

This procedure needs to check that the serial number is valid, and then update the database to indicate that the unit is active in table VLinkUnit. It must also log that the VLink has connected.

**Arguments**serialnumber varchar(50) VLink serial number

**Return**  
unitid int32 0 – if unit is not valid  
 > 0 = unit id number for subsequent calls into database

## proc\_disconnect

This procedure will update the VLinkUnit table to indicate that the unit is inactive. . It must also log that the VLink has disconnected.

**Arguments**unitid int32 unit id number returned from proc\_connect

**Return**  
none

## proc\_setproperty

This procedure will store a new value for a VLink unit into table VLinkProperties. The value will need to be stored with a timestamp. If the property ID is greater than 10000 than only a single instance of the property needs to be kept in the database (existing record with same property ID is erased). See Appendix A for property ID numbers.

**Arguments**unitid int32 unit id number returned from proc\_connect  
propertyid int32 identifier for property data (Appendix A)  
propertyvalue varchar(50) value for the property

**Return**  
status int32 0 = success, otherwise fail code

## proc\_getproperty

This procedure will return the value of a unit property, retrieved from VLinkProperties table. See Appendix A for property ID numbers.

**Arguments**unitid int32 unit id number returned from proc\_connect   
propertyid int32 identifier for property data (Appendix A)

**Return**  
value varchar(50) value for the property, this may be NULL

## proc\_getchannels

This procedure will return a dataset of channel information for currently configured channels. The channel configuration comes from VLinkChannels table. The return is a record set with all channel configurations, it may be null or empty string if there are no channels configured.

**Arguments**unitid int32 unit id number returned from proc\_connect

**Return**sensorid int32 VMC sensor ID number  
bustype int32 Bus Type (1=J1708, 2 = CAN, 3=Physical Sensor…)  
PGN\_PID int32 PGN or PID  
source int32 Source Address  
start int32 start location of data within message data  
size int32 size of data within message data

## proc\_getdataconversion

This procedure will return a single dataset for how to convert the raw data to engineering values.

**Arguments**unitid int32 unit id number returned from proc\_connect  
sensorid int32 VMC sensor ID number

**Return**type int16 conversion type enum  
value1 real conversion value #1  
value2 real conversion value #2

## proc\_storedatapacket

This procedure will return a status for storing a new data packet to the database.

**Arguments**unitid int32 unit id number returned from proc\_connect   
sensor\_id int32 sensor id number for this packet  
value1 int32 sensor value (for integer type channels)  
value2 real sensor value (for floating point type channels)  
value3 varchar(250) sensor value (for string type channels)  
packetdate datetime the timestamp for the packet

**Return**status int32 0 = success, otherwise fail code

# Configure Alarms

The next procedures allow you to modify the configuration of alarms.

## proc\_getalarmconfiguration

This procedure will return a dataset of alarms that must be checked for the unitID. The dataset is returned from the VLinkAlarmConfig table.

**Arguments**username varchar username\*  
unitid int32 unit id number returned from proc\_connect

**Return**id int32 unique identifier of the alarm in the database  
UnitID int32 VLink unit identifier  
AlarmID int32 enumerator for alarm type  
SensorID int32 VMC sensor enum  
LowLimit Real low limit for alarm  
HighLimit real high limit for alarm  
OnAction varchar action to take when alarm occurs  
OffAction varchar action to take when alarm ends  
Text varchar additional text to include with email/text  
LinkedAlarm int32 alarm index that must also be set

\*The username is only required if you are requesting the alarms for all the units, ie: unitid = -1.

## proc\_configurealarm

This procedure will modify an existing alarm configuration, or create new ones.

**Arguments**username varchar username, is required  
id int alarm id, can be -1 to create new alarms  
unitid int unit id, can be -1 to create new alarms for all units  
alarmid int alarm type, must be > 0  
sensorid int sensor channel to monitor, must be > 0  
low varchar optional low limit for sensor  
high varchar optional high limit for sensor  
onaction varchar optional action to take when alarm begins  
offaction varchar optional action to take when alarm ends  
text varchar optional additional text to send with alarm nofitication  
linked int optional id of alarm that must first be triggered before   
 this alarm can be set  
**Return**status int <0 is error, >0 is new id for this alarm

## proc\_deletealarmconfiguration

This procedure will delete one or more alarm configurations.

**Arguments**  
username varchar username, is required  
unitid int unit id, can be -1 to delete alarms for all units  
id int alarm id, can be -1 to delete all alarms

**Return**status int 0 = success, otherwise fail code

Modify Active Alarms  
The next 4 procedures all you to modify alarms that need to be set, or have already been set.

## proc\_setalarm

This procedure will record that an alarm has been raised, it returns an id value for the specific alarm.

**Arguments**unitid int32 unit id number returned from proc\_connect   
alarmid int32 alarm number from the alarm config table  
date datetime timestamp of when the alarm began  
value real value associated with the alarm  
text varchar(250) string associated with the alarm

**Return**status int32 > 0 = id for the new alarm, otherwise fail code

## proc\_endalarm

This procedure will record that an alarm is no longer active.

**Arguments**unitid int32 unit id number returned from proc\_connect   
id int32 id number returned from proc\_setalarm  
date datetime timestamp of when the alarm ended

**Return**status int32 0 = success, otherwise fail code

## proc\_getalarms

This procedure will return an array of alarms, you can define if you only want new alarms, and a time range

**Arguments**unitid int32 unit id number returned from proc\_connect   
new char 0 or 1 if you only want new alarms  
start datetime optional start timestamp  
end datetime optional end timestamp

**Return**id int unique alarm index  
UnitID int unit id number  
AlarmID int32 alarm type  
BeginDate datetime time when the alarm began  
EndDate datetime time when the alarm ended  
AlarmString varchar(255) string associated with the alarm  
AcknowledgeDate datetime time when the alarm was acknowledged  
AcknowledgeBy varchar(50) username of who acknowledged the alarm  
AlarmValue real recorded value that raised the alarm

## proc\_acknowledgealarm

This procedure is used to acknowledge an alarm

**Arguments**unitid int32 unit id number  
id int32 unique alarm index  
name varchar(50) name of person acknowledge

**Return**status int32 0 = success, otherwise fail code