



**Build
With
DataOne
Software Solutions**

VINBasic™

Technical Documentation: VINBasic™

DATAONE SOFTWARE | DIVISION OF DOMINION ENTERPRISES



DataOne VINBasic™

- Scope: 1981-present, consumer and light duty commercial vehicles
- US Market destination vehicles (although origin can be other than US)
- Technical: CSV file, delivered via FTP with weekly updates
 - Monday 4AM EST
 - Complete replacement file
- Implementation: Knowledge of simple database, query /look-up

The VINBasic™ file will be imported into a database platform of choice (MS Access, MYSQL, SQL Server, Oracle etc), where queries and lookups can be defined for use of the file.

VINBasic™ contains most commonly used fields including vehicle type, doors, dimensions and weights, engines, transmissions and drive types, as well as some industry specific fields such as restraint type, available colors, and vehicle MSRP (price as new).

USE OF VIN PATTERN

There is no need to "parse" out a VIN and associate values with each portion of the VIN such as the WMI (World Manufacturers Identification). This has already been done as to make it as easy as possible to VIN decode vehicles.

The VIN Pattern uses digits 1-8, skipping 9, and using digits 10 and 11 of the 17 digit VIN. For example, the VIN Pattern for the VIN 1GA2GYDG2A1107842 would be as follows:

1GA2GYDG|2|A1|107842* Resulting in: 1GA2GYDG||A1

VIN VALIDATION

Wherever users will be manually entering a VIN number, it is a good idea to perform some preliminary validation on the VIN number entered before attempting to query the VINBasic™ file. Below are some recommendations for VIN validation including checksum and illegal characters that are useful to implement.

Illegal Characters: The letters “I”, “O”, and “Q” are not used within a VIN system structure. If a VIN number is entered with these characters – this should be red flagged and identified to the user for correction.

Vehicle Checksum: Using the ninth digit of the VIN number, a calculation can be performed against the other digits of the VIN to verify whether the VIN is valid. For an excellent example of a VIN-check algorithm see the following URL for VINs at Wikipedia.org

[http://en.wikipedia.org/wiki/Vehicle Identification Number](http://en.wikipedia.org/wiki/Vehicle_Identification_Number)

OUT OF MARKET VIN DATA

DataOne's data primarily covers light-duty and passenger vehicles destined for sale in the United States. This includes passenger cars and light-duty pickup trucks up to and including F-450, Ram 3500, and Sierra 3500.

DataOne also collects limited data for vehicles outside the scope described above, including Canadian, Fleet, Chassis, and Specialty/Professional use vehicle data. This is referred to as "Out of Market" data, and can be added to the VINBasic™ file by contacting DataOne support at support@dataonesoftware.com

DataOne's Out of Market VIN data is designed to provide very basic vehicle identification for vehicles that are not covered in the primary collection of data. As such, many fields may appear blank for out of market vehicles. All Out of Market records are flagged in the VINBasic™ file using the "OUT_OF_MARKET" field. Please note that the coverage for these vehicles is not comprehensive, and that data is provided “as is” to further assist our Clients in identification of vehicles outside our standard data set.

MySQL Create Table Statement for VINBasic

Please note that this includes all available VINBasic™ fields. Lines representing fields that are not under subscription should be removed. If lines are deleted, it is important to note that there should be **no comma** on the second to last line (right before the closing parenthesis) to maintain the syntax of the statement.

MySQL Create Table Statement

```
CREATE TABLE IF NOT EXISTS `DataOne_VINBasic` (  
  `vin_pattern` varchar(10) NOT NULL,  
  `vehicle_id` int(10) unsigned NOT NULL,  
  `year` smallint(4) unsigned NOT NULL,  
  `make` varchar(24) NOT NULL,  
  `model` varchar(32) NOT NULL,  
  `trim` varchar(48) NOT NULL,  
  `style` varchar(128) NOT NULL,  
  `vehicle_type` varchar(24) NOT NULL,  
  `body_type` varchar(32) NOT NULL,  
  `body_subtype` varchar(32) NOT NULL,  
  `doors` tinyint(3) unsigned NOT NULL,  
  `msrp` int(10) unsigned NOT NULL,  
  `plant` varchar(32) NOT NULL,  
  `restraint_type` varchar(255) NOT NULL,  
  `gvw_range` varchar(20) NOT NULL,  
  `length` float NOT NULL,  
  `height` float NOT NULL,  
  `width` float NOT NULL,  
  `wheelbase` float NOT NULL,  
  `curb_weight` smallint(5) unsigned NOT NULL,  
  `gross_vehicle_weight_rating` int(11) NOT NULL,  
  `tmp_wheel_dia` varchar(12) NOT NULL,  
  `tmp_tank1_gal` varchar(6) NOT NULL,  
  `max_payload` mediumint(8) NOT NULL,  
  `def_engine_id` int(10) unsigned NOT NULL,
```

```
`drive_type` varchar(3) NOT NULL,  
`fuel_type` varchar(64) NOT NULL,  
`def_engine_block` varchar(1) NOT NULL,  
`def_engine_cylinders` tinyint(3) unsigned NOT NULL,  
`def_engine_size` float NOT NULL,  
`engine_size_uom` varchar(2) NOT NULL,  
`def_engine_aspiration` varchar(64) NOT NULL,  
`def_trans_id` int(10) unsigned NOT NULL,  
`def_trans_type` varchar(3) NOT NULL,  
`def_trans_speeds` tinyint(3) unsigned NOT NULL,  
`ext_color_name` text NOT NULL,  
`ext_mfr_color_name` text NOT NULL,  
`ext_mfr_color_code` text NOT NULL,  
`ext_r_code` text NOT NULL,  
`ext_g_code` text NOT NULL,  
`ext_b_code` text NOT NULL,  
`int_color_name` text NOT NULL,  
`int_mfr_color_name` text NOT NULL,  
`int_mfr_color_code` text NOT NULL,  
`int_r_code` text NOT NULL,  
`int_g_code` text NOT NULL,  
`int_b_code` text NOT NULL  
) ENGINE=MyISAM;
```