Addon DB compiler

ver20161220

Generated by Doxygen 1.8.11

Contents

Chapter 1

Add on DB compiler MAIN PAGE

Page for search target data

1.1 Working Platform

The working platform is Linux

1.2 command line option

-m import mid file
-d parse db file

Chapter 2

Page for search target data

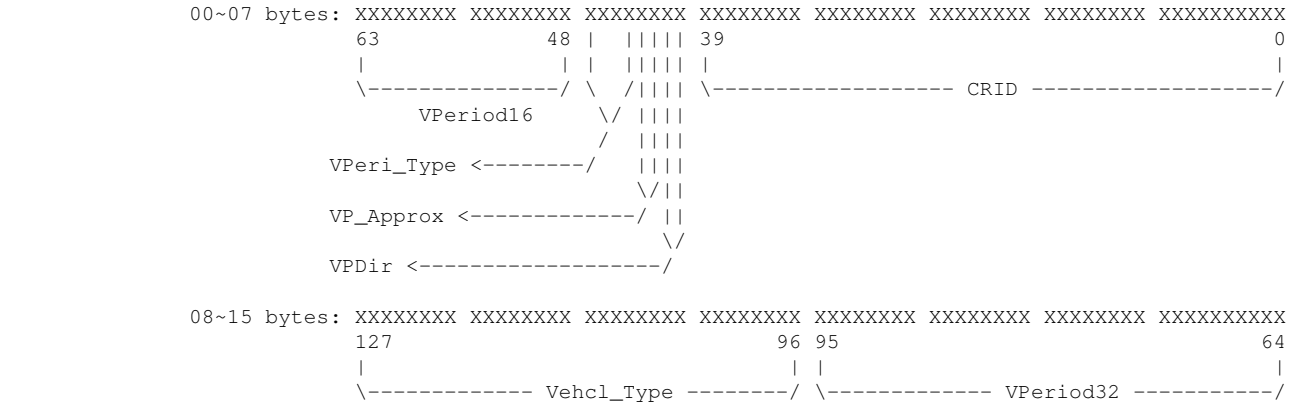
This page introduce how to search target data such as CR, toll info or HW info.

2.1 reach the CR info

2.1.1 CR info binary format

The CR info is up to 16 bytes.

The bit sequence is like following.



2.1.2 field detail

1. bit 0∼39, 40 bits : CRID
2. bit 40∼41, 2 bits : VPDir, indicate the restrict direction. The value range is [0, 3].

    - case value == 0 : Default value. The meaning is no direction.
    - case value == 1 : Restriction on the bidirections.
    - case value == 2 : Restriction on the order direction.
    - case value == 3 : Restriction on the reverse direction.

3. bit 42∼43, 2 bits : VP_Approx, the value indacates the aproximation for VPeriod. The value ranges as [0, 2].
   - case value == 0 : Not available.
   - case value == 1 : Not the value approximated.
   - case value == 2 : The value is approximated.
   - case value == 3 : Not defined.
4. bit 44∼47, 4 bits : VPeri_Type, value range [0,2].
   - case value == 0 : not available
   - case value == 1 : MMdd V.S. hhmm
   - case value == 2 : hhmm V.S. t1∼t7
   - case other value: not defined
5. bit 48∼95, 48 bits: VPeriod, the bit sequence rules the time of traffic regulation.
   - case VPeri_Type == 1 :
     - the part VPeriod16 : bit 48∼63, 16 bits.
       (a) bit 48∼51, 4 bits, start month. The value ranges [0, 12].
           * case value == 0 : not available.
           * case value == 1..12 : Jan .. Dec.
           * case other value : not defined.
       (b) bit 52∼55, 4 bits, end month. The value ranges [0, 12].
           * case value == 0 : not available.
           * case value == 1..12 : Jan .. Dec.
           * case other value : not defined.
       (c) bit 56∼63, 8 bits, not defined.
     - the part VPeriod32 : bit 64∼95, 32 bits.
       (a) bit 64∼68, 5 bits, start date. The value ranges [0, 31].
           * case value == 0 : not available.
           * case value == 1..31 : The day in a month.
       (b) bit 69∼73, 5 bits, end date. The value ranges [0, 31].
           * case value == 0 : not available.
           * case value == 1..31 : The day in a month.
       (c) bit 74∼78, 5 bits, start hour. The value ranges [0, 23].
           * case value == 0..23 : hour.
           * case the other value : not available.
       (d) bit 79∼83, 5 bits, end hour. The value ranges [0, 23].
           * case value == 0..23 : hour.
           * case the other value : not available.
       (e) bit 84∼89, 6 bits, start miniute. The value ranges [0, 59].
           * case value == 0..59 : minute.
           * case the other value : not available.
       (f) bit 90∼95, 6 bits, start miniute. The value ranges [0, 59].
           * case value == 0..59 : minute.
           * case the other value : not available.
   - case VPeri_Type == 2 :
     - the part VPeriod16 : bit 48∼63, 16 bits. Not defined.
     - the part VPeriod32 : bit 64∼95, 32 bits.
       (a) bit 64∼70, 7 bits, bit sequence.
           * bit 0..6 : Sun, Mon..Sat.

(b) bit 71∼73, 3 bits, not defined.
(c) bit 74∼78, 5 bits, start hour. The value ranges [0, 23].
* case value == 0..23 : hour.
* case the other value : not available.
(d) bit 79∼83, 5 bits, end hour. The value ranges [0, 23].
* case value == 0..23 : hour.
* case the other value : not available.
(e) bit 84∼89, 6 bits, start miniute. The value ranges [0, 59].
* case value == 0..59 : minute.
* case the other value : not available.
(f) bit 90∼95, 6 bits, start miniute. The value ranges [0, 59].
* case value == 0..59 : minute.
* case the other value : not available.

- case VPeri_Type == 3 :
  - the part VPeriod16 : bit 48∼63, 16 bits. Not defined.
  - the part VPeriod32 : bit 64∼95, 32 bits.
    (a) bit 64∼73, 10 bits, not defined.
    (b) bit 74∼78, 5 bits, start hour. The value ranges [0, 23].
    * case value == 0..23 : hour.
    * case the other value : not available.
    (c) bit 79∼83, 5 bits, end hour. The value ranges [0, 23].
    * case value == 0..23 : hour.
    * case the other value : not available.
    (d) bit 84∼89, 6 bits, start miniute. The value ranges [0, 59].
    * case value == 0..59 : minute.
    * case the other value : not available.
    (e) bit 90∼95, 6 bits, start miniute. The value ranges [0, 59].
    * case value == 0..59 : minute.
    * case the other value : not available.

2.2 reach the Toll info

- The ETA info One toll ETA info is up to 8 bytes. The fields in one toll ETA info is like following.
  1. bit 00∼39, 40 bits : condion ID.
  2. bit 40∼43, 4 bits : toll type.
  3. bit 44∼47, 4 bits : lane number.
  4. bit 48∼63, 16 bits : not defined.
- The Pattern Info One toll pattern info is up to 16 bytes. The fields in one toll pattern info is like following.
  1. bit 00∼39, 40 bits : condion ID.
  2. bit 40∼63, 24 bits : not defined.
  3. bit 64∼95, 32 bits : pattern number.
  4. bit 96∼127, 32 bits : arrow nubmer.

2.3 reach the HW Junction info

One HW Junction info is up to 24 bytes and has 3 parts. The part 1 is 8 bytes. The part 2 is 6 bytes and the part 3 is 10 bytes.

the fields in these files are like following.

- part 1 : 8 bytes.
    1. bit 00∼39, 40 bits : ID
    2. bit 40∼47, 8 bits : not defined.
    3. bit 48∼51, 4 bits : access type.
    4. bit 52∼55, 4 bits : attribute.
    5. bit 56∼63, 8 bits : estab item.
        - bit 56 : 1, restaurant. 0 , no restaurant available.
        - bit 57 : 1, shop. 0, no shop available.
        - bit 58 : 1, inn. 0, no inn available.
        - bit 59 : 1, public toilet. 0, no public toilet available.
        - bit 60∼63 : Natural number, the gasoline station.
            * value == 21 : PetreChina.
            * value == 22 : sinopec
            * value == 23 : shell
            * value == 24 : Mobil
            * value == 25 : Brithish Petroleum
            * value == 26 : other
- part 2 : 6 bytes. bit 00∼39, 40 bits : NodeID bit 40∼63, 16 bits : not defined.
- part 3 : 10 bytes.
    1. byte 0∼4 : inLinkID.
    2. byte 5∼9 : outLinkID.

Chapter 3

sqlite wrapper

some detail for sqlite wrapper

Chapter 4

Module Index

4.1 Modules

Here is a list of all modules:

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 6

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 7

Module Documentation

7.1 db group

something about CM db

something about CM db

some detail about CM db

Chapter 8

Class Documentation

8.1 CCmDatabase Class Reference

compiler Database

`#include <cm_db.hpp>`

Public Member Functions

- bool import_mid (const char *)
  import ".mid" files*
- bool **parse_db** (const char *)

8.1.1 Detailed Description

compiler Database

8.1.2 Member Function Documentation

8.1.2.1 bool CCmDatabase::import_mid (const char * *path*)

import "*.mid" files

Parameters

path	the mid file path. The mid file base name should be leading by Toll_ETA, Toll_Patern

Returns

import OK or not

Return values

true	import OK
false	import failed

The documentation for this class was generated from the following files:

- /home/wangxl/mylab/codes/compiler/addon/inc/cm_db.hpp
- /home/wangxl/mylab/codes/compiler/addon/src/cm_db.cpp

8.2 CCmSqlite Class Reference

Classes

- class statement

Public Member Functions

- **CCmSqlite** (const char *)
- statement * **create_statement** (const char *)
- void **remove_statement** (statement *)
- bool **execute** (const char *)
- bool **backup** (const char *)

The documentation for this class was generated from the following files:

- /home/wangxl/mylab/codes/compiler/addon/inc/cm_sqlite.hpp
- /home/wangxl/mylab/codes/compiler/addon/src/cm_sqlite.cpp

8.3 CCmSqlite::statement Class Reference

Public Member Functions

- **statement** (sqlite3 *, const char *)
- bool **step** ()
- bool **step_row** ()
- const char * **get_text** (size_t)
- bool **bind_text** (size_t, const char *)
- void **reset** ()

The documentation for this class was generated from the following files:

- /home/wangxl/mylab/codes/compiler/addon/inc/cm_sqlite.hpp
- /home/wangxl/mylab/codes/compiler/addon/src/cm_sqlite.cpp

Chapter 9

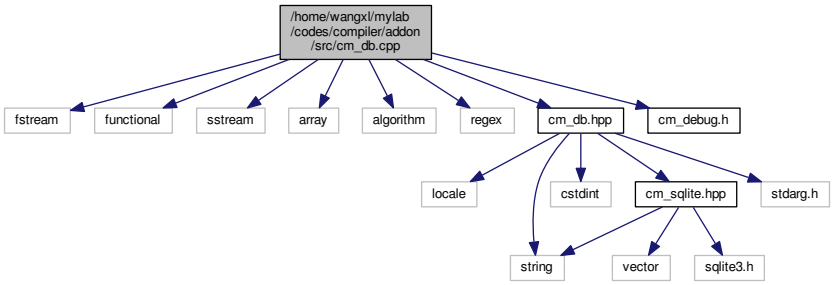
File Documentation

9.1 /home/wangxl/mylab/codes/compiler/addon/src/cm_db.cpp File Reference

the database implement for compiler

```
#include <fstream>
#include <functional>
#include <sstream>
#include <array>
#include <algorithm>
#include <regex>
#include "cm_db.hpp"
#include "cm_debug.h"
```

Include dependency graph for cm_db.cpp:



Macros

- #define **LOG_HEADER** "[CM_DB]"
- #define **TABLE_CR** "CR"
- #define **TABLE_Toll_ETA** "Toll_ETA"
- #define **TABLE_Toll_Pattern** "Toll_Pattern"
- #define **TABLE_HW_Junction** "HW_Junction"

Functions

- uint32_t **_stou32** (const std::string s, int base=10)
- uint64_t **_stou64** (const std::string s, int base=10)

9.1.1 Detailed Description

the database implement for compiler

the class routine things

Author

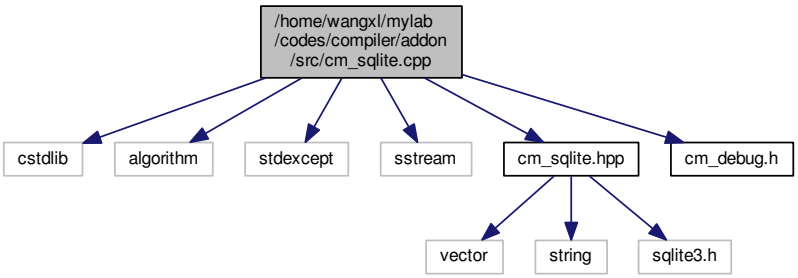
Wang Xiaolong (WXL), wangxl3@mapbar.com

9.2 /home/wangxl/mylab/codes/compiler/addon/src/cm_sqlite.cpp File Reference

sqilte wrapper implement

```
#include <cstdlib>
#include <algorithm>
#include <stdexcept>
#include <sstream>
#include "cm_sqlite.hpp"
#include "cm_debug.h"
```

Include dependency graph for cm_sqlite.cpp:



Macros

- #define **LOG_HEADER** "[CM_SQLITE]"

9.2.1 Detailed Description

sqilte wrapper implement

wrapper create table, statement step etc.

Author

Wang Xiaolong (WXL), wangxl3@mapbar.com

Index