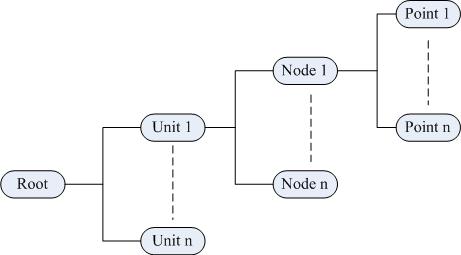
# openPlant数据结构

实时数据库处理的主要对象为从现场各控制系统采集来的实时数据，为了管理这些数据，保证数据的唯一性，openPlant实时数据库采用了以系统来划分的层次结构，整个系统采用树状结构，共分为RTDB、node、point三层，如下图所示：



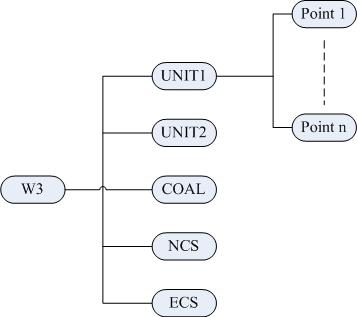
root 系统的根节点

RTDB 可表示一个厂的数据库实例。如一个集团下属多个分厂，在每个分厂安装有一个openPlant数据库，那么每个RTDB 表示一个分厂。对于只有一个数据库实例的目前缺省名称为W3。

node 可以表示一个厂中的某个系统，如DCS、输煤、化学、ECS、NCS等，或用户自定义的每个节点，如计算点的节点、手工输入点的节点等。

point 表示某个采集节点中的点。

例如：一个只装有单一openPlant实时数据库的厂，该厂有1、2号机组DCS，输煤、NCS、ECS等控制系统，则该厂的数据结构如下



如1号机组中有个点名为TE0001，则该点在系统中的全局点名为

W3.UNIT1.TE0001

如NCS系统中有个点名为AI0001，则该点在系统中的全局点名为

W3.NCS.AI0001

# 数据表结构

实时数据库主要注重对实时数据的高效处理，实时数据库在实际应用中通常看作是由如下几张表格构成的:

点表：Point

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 点标识 |
| UD | long | UUID |
| ND | int | 父节点标识 |
| PT | int | 点的来源 |
| RT | int | 点的类型 |
| PN | string | 点名 |
| AN | string | 别名 |
| ED | string | 描述 |
| KR | string | 特征字 |
| SG | byte | 安全组 |
| FQ | short | 分辨率 |
| CP | short | 处理器 |
| HW | int | 模块地址 |
| BP | short | 通道号 |
| LC | byte | 报警类型 |
| AP | byte | 报警优先级 |
| AR | byte | 存档 |
| FL | int | 标志位 |
| ST | string | 值为1时的描述 |
| RS | string | 值为0时的描述 |
| EU | string | 单位 |
| FM | short | 显示小数位 |
| IV | float | 初始值 |
| TV | float | 量程上限 |
| BV | float | 量程下限 |
| LL | float | 报警低限 |
| HL | float | 报警高限 |
| ZL | float | 报警低2限 |
| ZH | float | 报警高2限 |
| DB | float | 死区 |
| DT | byte | 死区类型 |
| KZ | byte | 压缩类型 |
| KT | byte | 计算类型 |
| CT | datetime | 修改时间 |
| EX | string | 计算表达式 |
| GN | string | 全局名称 |

**节点表：Node**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 点标识 |
| UD | long | UUID |
| ND | int | 父节点标识 |
| PN | string | 名称 |
| ED | string | 描述 |
| FQ | int | 分辨率 |
| LC | int | 报警类型 |
| AR | string | 存档 |
| OF | string | 离线 |
| CT | datetime | 修改时间 |
| GN | string | 全局名称 |

**实时表：Realtime**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 测点ID |
| GN | string | 测点名称，如W3.TEST.AX001 |
| TM | datetime | 测点更新时间，从1970-1-1到当前的秒数 |
| DS | short | 测点状态 |
| AV | double | 测点数值 |

**历史表：Archive**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 测点ID |
| GN | string | 测点名称，如W3.TEST.AX001 |
| TM | datetime | 测点更新时间，从1970-1-1到当前的秒数 |
| DS | short | 测点状态 |
| AV | double | 测点数值 |
| MODE | string | raw,span,plot,flow,max,min,avg,mean,sum,stdev,arch |
| INTERVAL | string | 时间间隔：1h,1m,1s |
| QTYPE | int | 质量过滤：0-不过滤，1-去除坏点, 2-去除超时，3-去除1和2 |

**历史统计表：Stat**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 测点ID |
| GN | string | 测点名称，如W3.TEST.AX001 |
| TM | datetime | 测点更新时间，从1970-1-1到当前的秒数 |
| DS | short | 测点状态 |
| FLOW | double | 累积值 |
| AVGV | double | 时均平均值 |
| MAXV | double | 最大值 |
| MINV | double | 最小值 |
| MAXTIME | datetime | 最大值时间 |
| MINTIME | datetime | 最小值时间 |
| INTERVAL | string | 时间间隔：1h,1m,1s |
| QTYPE | int | 质量过滤：0-不过滤，1-去除坏点, 2-去除超时，3-去除1和2 |

**报警表：Alarm（实时），AAlarm（历史）**

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| ID | int | 测点ID |
| GN | string | 测点名称，如W3.TEST.AX001 |
| RT | byte | 测点类型：AX,DX,I2,I4,R8 |
| AL | byte | 报警优先级 |
| AC | int | 报警颜色 |
| TF | datetime | 首次报警时间 |
| TA | datetime | 报警时间 |
| TM | datetime | 测点更新时间，从1970-1-1到当前的秒数 |
| DS | short | 测点状态 |
| AV | object | 测点数值 |

# 数据库表实现

OPIO(C#)主要针对数据库表支持的增删改查执行操作，接口简单易用。数据库表支持的操作如下，对应的字段属性参考第二部分 数据库表结构。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表** | **描述** | **Select** | **Insert** | **Update** | **Delete** |
| Database | 数据库表 | √ | √ | √ | √ |
| Node | 节点表 | √ | √ | √ | √ |
| Point | 点表 | √ | √ | √ | √ |
| Realtime | 实时数据表 | √ | √ | √ | × |
| Archive | 历史数据表 | √ | √ | √ | √ |
| Stat | 历史统计表 | √ | × | × | × |
| Alarm | 实时报警表 | √ | × | × | × |
| AAlarm | 历史报警表 | √ | × | × | × |

# OPIO(C#)函数接口

|  |  |  |
| --- | --- | --- |
| **函数功能** | **函数接口** | **参数说明** |
| 建立连接 | con = new OPAPI.Connect(string ip,int port,int timeout,string user, string password) | 参数：  ip地址  port端口号  timeout连接超时时间  user用户名  password密码  返回值：连接对象 |
| 数据库连接状态判断 | con.isAlive() |  |
| 关闭连接 | con.close() |  |
| SQL(增、删、改、查) | resultSet=con.executeQuery(sql) | 参数：  SQL语句  返回值：结果对象 |
| OPIO(插入) | resultSet=con.insert(string tableName,string []colNames,object[,]rows, Dictionary<string, object> options) | 参数：  tableName表名  colNames字段名  rows插入数据  options可选参数  返回值：结果对象 |
| OPIO(删除) | resultSet=con.delete(string tableName,string[] colNames,object[] keys) | 参数：  tableName表名  colNames字段名  keys关键字(ID或GN)  返回值：结果对象 |
| OPIO(修改) | resultSet=con.update(string tableName,string[]colNames,object[,]rows) | 参数：  tableName表名  colNames字段名  rows插入数据  返回值：结果对象 |
| OPIO(查询) | resultSet=con.select(string tableName,string[]colNames,object[]keys, Dictionary<string, object> options) | 参数：  tableName表名  colNames字段名  keys关键字(ID或GN)  options可选参数  返回值：结果对象 |
| 移动游标 | bNext=resultSet.next() | 返回值：下移游标是否成功 |
| 订阅 | ahObject=con.openAsync(string tableName, event\_cb cb, object[]keys) | 参数：  tableName表名  cb回调函数  keys关键字  返回值：结果对象 |
| 动态订阅 | ahObject.add(object[]keys) | 参数：  keys关键字 |
| ahOject.remove(object[]keys) | 参数：  keys关键字 |
| 获取数据 | colN=resultSet.columnLabel(int column) | 参数：column列号  返回值：列名字 |
| colV=resultSet.getObject(int colNum) | 参数：  keyOrColumn字段名或者列号  返回值：对应内容 |
| 其他获取对应类型接口见OPAPI中resultSet对象 | |
| 释放内存 | resultSet.close() |  |

# 函数示例

具体使用示例请参考OPIO(C#) demo示例。