

M4DB(多主机及数据库管理与监控系统)用户手册

文档	版本	编制日期
用户手册	V2.0	2016/6/1
	V2.1	2018/1/12
	V2.1	2020/5/1

目录

M4DB(多主机及数据库管理与监控系统)用户手册	1
1. 引言	4
1.1 编写目的.....	4
1.2 定义.....	4
2. 软件概述	4
2.1 功能.....	4
2.2 模块介绍.....	4
2.3 主要特点.....	5
3. 安装配置	5
3.1 系统需求.....	5
3.2 安装.....	6
3.3 数据库配置.....	6
3.4 运行.....	8
4. 操作说明	9
4.1. 客户端软件.....	9
4.1.1) 登录界面.....	9
4.1.2) 主界面.....	9
4.1.3) 配置 -> 数据库.....	10
4.1.4) 配置 -> 主机.....	11
4.1.5) 配置 -> 应用服务.....	12
4.1.6) 配置 -> 网络设备.....	12
4.1.7) 配置 -> 管理组.....	13
4.1.8) 配置 -> 监控对象.....	14
4.1.9) 配置 -> 产品信息.....	16
4.1.10) 配置 -> 程序选项.....	17
4.1.11) 配置 -> 用户管理.....	18
4.1.12) 配置 -> 机器扩展属性.....	18
4.1.13) 配置 -> 刷新管理列表.....	19
4.1.14) 管理工具 -> 新脚本.....	19
4.1.15) 管理工具 -> 运行命令.....	19
4.1.16) 管理工具 -> 执行脚本文件.....	20
4.1.17) 管理工具 -> 输出结果比较/导出	20
4.1.18) 管理工具 -> 关闭输出窗口.....	21
4.1.19) 管理工具 -> 上传脚本到主机.....	21
4.1.20) 管理工具 -> 导入/导出监控预定义	22
4.1.21) 管理工具 -> SNMP-MIB 查询.....	22
4.1.22) 管理工具 -> JMX-MBEAN 查询	24
4.1.23) 监控 -> 开始监控.....	24
4.1.24) 监控 -> 停止监控.....	24
4.1.25) 监控 -> 检查监控状态.....	24
4.1.26) 监控 -> 清除缓存对象.....	25

4.1.27)	监控 -> 导出全部监控结果.....	25
4.1.28)	监控 -> 监控结果展示.....	25
4.1.29)	帮助.....	25
4.2.	监控结果展示.....	25
4.2.1	实时监控.....	27
4.2.2	历史数据.....	32
4.2.3	预警及处理.....	36
4.2.4	拓扑图定义.....	38
4.3.	监控调度模块.....	43
4.4.	消息发送模块.....	44
4.5.	代理模块.....	46
5.	监控对象定义	47
5.1	SSH/TELNET 连接.....	47
5.1.1)	执行脚本的定义.....	48
5.1.2)	结果解析的定义.....	48
5.1.3)	项目定义.....	49
5.1.4)	表达式定义.....	50
5.2	JDBC 连接.....	51
5.3	SNMP 连接.....	52
5.4	JMX 连接	54
5.5	HTTP 连接	57
5.6	IPMI 连接	58
5.7	SOCKET 连接	60
5.8	代理连接.....	60

1. 引言

1.1 编写目的

本文档描述了 M4DB 产品的功能，安装、使用说明等，供使用软件的操作人员阅读。

1.2 定义

M4DB 是 Manage Monitor Multiple Machine & Database 的缩写，是多主机及数据库管理与监控系统。版本 V2.0 后支持了 SNMP, JMX, HTTP, SOCKET, IPMI 协议，可以实现对网络设备，应用服务等监控。

- 监控源 所有被监控的主机、数据库、网络设备、应用服务统称为监控源
- 监控对象 监控源下定义的监控，如文件系统使用、数据库日志信息等
- 监控项目 监控对象的明细信息，如文件系统使用对象包含项目：文件系统名、总空间、可用空间等

2. 软件概述

2.1. 功能

M4DB 是一款监控及运维管理产品，通过 SSH/TELNET, JDBC, SNMP, JMX, HTTP, SOCKET, IPMI 等方式实现对主机、数据库、网络设备、应用服务的监控与运维。通过定时调度采集系统信息、业务数据等，经过解析评估实现监控目的。客户端软件完成监控定义、自动化批量运维等功能。

在监控方面：M4DB 产品有灵活、简便易用的特点，能处理很多有特别需求的监控对象；在监控的结果输出方面，M4DB 有详尽的输出内容，基本能从监控展示界面中反映故障发生的原因；保留的历史数据，提供查询分析功能，能在一张图上反映多个监控对象的趋势图；

在自动化批量运维方面：M4DB 是完全的图形化界面，可选择批量运维的对象，支持主机、数据库、网络等多种监控对象的批量操作，输出结果展示在不同的 tab 页上。

2.2. 模块介绍

M4DB 产品主要包含 5 个功能模块：

- 客户端软件 – m4dbApp

主要完成主机、数据库等监控源定义、监控对象定义等，以及对主机、数据库等的自动化批量运维；监控模块的状态查询等功能。

- 监控展示 – m4web

Web 系统展示监控采集结果、告警处理、历史数据的查询分析，以及拓扑图定义功能。

- 监控调度 – monitor

该模块以 2 分钟间隔进行调度定义的监控对象，采集监控内容，进行解析评估，写入到数据库中。

- 消息发送 – m4message

消息发送模块通过接口实现，将产生的告警消息发送给用户。内置有邮件发送模块。

- 代理 - m4agent

一些无法通过直接连接方式获取监控信息的系统，通过部署代理端完成监控信息采集工作。

2.3. 主要特点

M4DB 系统在监控、运维方面有如下的几个特点：

- 简单、灵活的监控对象定义

系统提供的 ssh/telnet, jdbc, snmp, jmx, http, socket, ipmi 以及代理等多种连接方式，基本上可以涵盖大部分的监控实体。基于脚本、解析、评估方式的监控采集，能解决各种自定义监控需求，而无需二次开发。

监控定义可以批量复制、多个监控源同时批量定义、导入导出定义等功能，极大的简化定义工作。

- 并发批量运维

客户端软件提供了图形化的、并发、批量的脚本执行功能。可以并发地在多个监控源执行相同脚本，或不同的脚本。执行过程可见（包括状态：not_connect, connected, running, completed ...，输出），并可随时中断执行。执行的状态结果展示在主输出窗口，各监控源的明细输出产生不同的 TAB 输出窗口，错误信息也在各自的 TAB 输出窗口中。

- 明细的监控结果输出

监控结果展示中的输出内容包含脚本执行的过程输出，发生执行错误、连接错误时，原始输出中也包含这些内容，能很清晰地反应故障原因，不用再次登录到监控源上查看具体原因。

- 历史数据分析

监控结果输出保留在历史数据中，不同监控对象可以定义不同的保留时间。历史数据可以查询，导出。监控对象定义的数字类型项目，都可以使用图表方式展示。不同监控对象的输出图表可以融合在一张图上展示。

3. 安装配置

3.1 系统需求

操作系统支持 windows, linux, unix 等系统；

数据库支持常见的关系型数据库，如 `oracle`, `mysql`, `derby`（内置）等；
Java 要求 1.6 以上版本。如果监控源使用了 `openssh7` 以上版本，`m4db` 使用 `ssh` 连接方式会产生异常，Java 可以使用 1.8 版本解决。
存储空间大小要求，至少 1GB 以上。定义的监控对象数量，监控输出保留的天数等都影响需要使用的空间。

3.2 安装

解压缩 `m4db.rar` 或 `m4db.zip` 到任意目录中。

目录结构如下：

- | - `agent` 代理端程序
- | - `bin` 客户端、监控执行程序等
 - | - `lib` jar 库文件
 - | - `config` 配置文件、文档
 - | - `m4config.xml` 程序选项(客户端工具使用):浏览器信息,监控程序,license
 - | - `m4applicationctx.xml` spring 架构配置文件，包含数据库连接信息
 - | - `monitor.log4j` 监控程序日志配置
 - | - `m4message.xml` 消息输出配置
 - | - `tables-XXX.sql init.sql` 不同数据库建表、初始化脚本
 - | - `rtxSender.java` 消息发送接口实例
 - | - `mibs` 目录 `snmp` 基本 MIB
- | - `db` 内置 `derby` 数据库
- | - `log` 日志输出，及客户端工具导出内容存放目录
- | - `web` web 应用 默认端口 8200, 地址 <http://127.0.0.1:8200/m4web>

`Bin` 目录下包含执行命令， `m4db.bat` 客户端软件

`monitor.sh / monitor.bat` 监控模块运行

`msgcfg.bat` 配置消息发送

`startmsg.bat` 消息发送模块

`startdb.sh / startdb.bat` 启动内置的 `derby` 数据库

`startweb.sh / startweb.sh` 启动 `web` 模块

3.3 数据库配置

目前测试通过的数据库包含 `derby` , `mysql`, `oracle` 等。使用其他数据库，将相应的 `jdbc` 驱动 jar 包复制到 `lib` 目录下，配置 `jdbc` 的连接信息。

使用 `jdbc` 连接的配置文件有 2 处：

`Config/m4applicationctx.xml` -- 客户端软件，监控模块使用

`Web/webapps/m4web/WEB-INF/classes/m4applicationctx.xml` – `m4web` 模块

选择数据库后，执行 `config/table-xxx.sql` 脚本文件，创建表， 执行 `init.sql` 文件插入初始数据（主要包含字典信息、预定义的产品、监控对象等）；

配置文件使用的是 spring 架构， 数据源 bean id : m4DataSource, 持久化使用的 hibernate 3, Bean id : mySessionFactory。
如 mysql 连接配置：

```
<bean          id="m4DataSource"          class="org.apache.commons.dbcp.BasicDataSource"
destroy-method="close">
    <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
    <property name="url" value="jdbc:mysql://127.0.0.1:3306/mydb?characterEncoding=GBK"/>
    <property name="username" value="sa"/>
    <property name="password" value="sa123"/>
</bean>
<bean                                     id="mySessionFactory"
class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
    <property name="dataSource" ref="m4DataSource"/>
    <property name="packagesToScan" value="com.m4db.entity"/>
    <property name="hibernateProperties">
        <value>
            hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
            hibernate.show_sql=false
            hibernate.jdbc.fetch_size=100
            hibernate.jdbc.batch_size=100
        </value>
    </property>
</bean>
```

采用 Oracle 数据库的配置：

```
<bean          id="m4DataSource"          class="org.apache.commons.dbcp.BasicDataSource"
destroy-method="close">
    <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@127.0.0.1:1521:mydb"/>
    <property name="username" value="sa"/>
    <property name="password" value="sa123"/>
</bean>
<bean                                     id="mySessionFactory"
class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
    <property name="dataSource" ref="m4DataSource"/>
    <property name="packagesToScan" value="com.m4db.entity"/>
    <property name="hibernateProperties">
        <value>
            hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
            hibernate.show_sql=false
            hibernate.jdbc.fetch_size=100
            hibernate.jdbc.batch_size=100
        </value>
    </property>
</bean>
```

采用其他数据库连接，配置类似，替换 driverclassname, url , hibernate.dialect

！注意 oracle 数据库中， timestamp 字段类型，通过 jdbc 传入的 date 类型数据访问时无法使用索引，需要修改监控模块下的配置文件 m4dbapplicationctx.xml，添加 bean querySqlConfig, 将查询 executetime 字段的语句转为 date 类型:

```
<bean          id="querySqlConfig"          class="com.m4db.context.M4QuerySqlConfig"
depends-on="m4DataSource">
    <property name="itemHistSql"><value>select b.catname, a.* from m4outputitemhistory a,
m4hostinfo b where a.hostname=b.hostname and a.hostname=? and a.objectname=? and
a.executetime=cast( ? as date) </value>
    </property>
    <property name="resultHistSql"><value>select b.catname, a.* from m4outputhistory a,
m4hostinfo b where a.hostname=b.hostname and a.hostname=? and a.objectname=? and
a.executetime=cast(? as date) </value>
    </property>
    <property name="whereStartExecuteSql"><value> and a.executetime >= cast(? as date)
</value></property>
    <property name="whereEndExecuteSql"><value> and a.executetime <= cast(? as date)
</value></property>
    <property name="delHistSqlCond"><value> and hostname=:hostname and
objectname=:objectname and executetime=cast(:executetime as date) </value></property>
</bean>
```

3.4 运行

运行监控模块、客户端软件前需要保证数据库可访问。

日常监控、告警需要执行

- monitor.sh / monitor.bat 命令，开始运行监控采集；
- Startmsg.sh / startmsg.bat 命令，消息发送。
- M4db.bat 客户端软件，进行定义配置、自动化批量运维
- Startweb.sh / startweb.bat 启动 tomcat 服务 m4web
使用不同的 web 访问端口，修改 web/conf/server.xml 文件中：<Connector port="8200"
处。

监控模块、web 模块、消息发送模块都可以单独部署在不同的机器上。

一般监控模块和 web 模块部署在同一台机器，消息发送模块部署在内、外网都访问的环境中，便于短信、邮件发送。

4. 操作说明

4.1. 客户端软件

客户端软件是进行配置、定义、自动化批量运维等工作的图形界面软件。需要安装在每个用户的本机中。初始登录的用户是 `admin`，口令: `adm123`。

M4db.bat / m4db.sh 命令中的内容为:

Java -Dm4db.home=.. -jar lib\m4dbApp.jar

可以在命令最后加参数,指向使用连接配置的 m4applicationctx.xml 文件,连接到不同的 m4db 系统。

注意: 监控调度模块在启动时缓存了需要采集信息的监控对象定义,所有配置的监控源、对象等发生变化,需要重启监控调度模块,或者通过 监控->清除缓存对象,使其生效或失效。

4.1.1) 登录界面

- 登录: 使用用户、口令登入到客户端软件。
- 取消: 进行基础配置,定义 jdbc 连接等。




4.1.2) 主界面

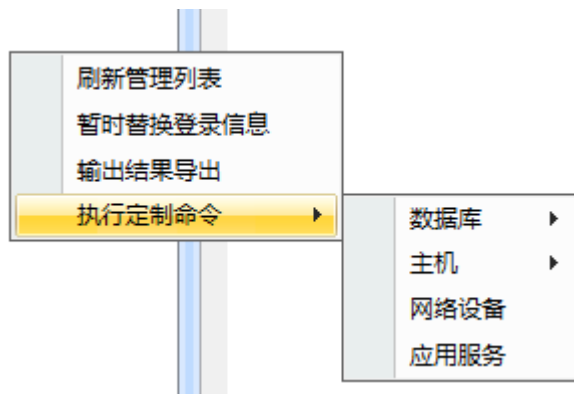
主界面是可拖动, 缩放的框架, 主要包含以下几个部分:

1. 菜单: 配置、管理工具、监控、帮助
2. 快捷按钮: 打开新的脚本窗口、执行脚本、终止执行、打开监控展示(m4web)
3. 可管理的对象列表树, 分 4 大类: 数据库、主机、网络设备、应用服务。

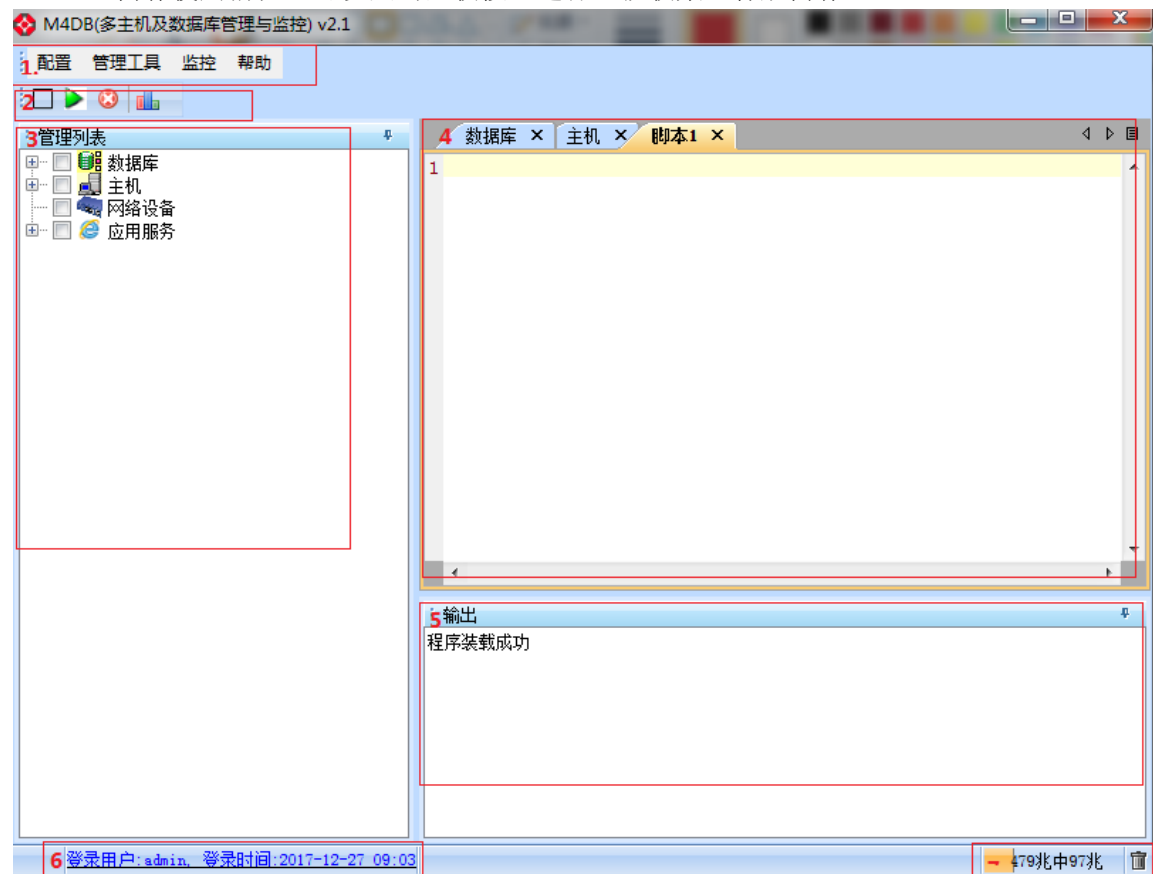
右键弹出菜单:

- 刷新管理列表 监控源、管理组等发生变化后, 重新载入列表
- 暂时替换登录信息 对选中的监控源, 使用临时的用户、口令登录
- 输出结果导出 把输出窗口产生的多个输出结果导出为 excel 或 text

- 执行定制命令 监控对象定义为  作为定制命令执行的, 可以批量执行, 不需要填写脚本



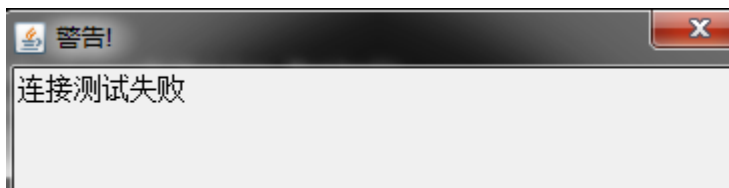
4. 多窗口的操作区
5. 操作的输出、报错提示等。输出窗口包括：
 - 主输出窗口 -- 不可被关闭；
 - 结果输出窗口 -- 以监控源的名称命名；
 - 错误输出窗口 -- 以监控源名称#Error# 命名；
 窗口内容包括两种： 文本形式， 表格形式。
 可以通过右键弹出菜单，可以对当前输出结果导出为 csv, html, excel 等
6. 登录用户信息， 可以修改当前用户口令
7. Java 内存使用情况，可以点击回收按钮进行垃圾收集，释放内存



4.1.3) 配置 -> 数据库

完成对需要监控的数据库的增、删、改操作。

- 测试连接：进行数据库连接测试，异常会弹出窗口



详细信息可查看 dos 窗口输出：

```

CA 管理员: 命令提示符 - m4db
com.mysql.jdbc.exceptions.jdbc4.CommunicationsException: Communications link failure

Last packet sent to the server was 0 ms ago.
    at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
    at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:52)
    at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:46)
    at java.lang.reflect.Constructor.newInstance(Constructor.java:525)
  
```

- 机器名：要保证唯一性
- 产品类型：列表选择，需要在配置 -> 产品信息 处定义
- URL：选择不同的产品类型会自动填充 URL，如果不同可以手工修改
- 启用：反选后，该数据库下所有定义的监控对象将不再收集信息。（需要重启监控调度，或者点击 监控->清除缓存对象）
- 通过代理：在被监控的数据库主机上安装代理 m4agent，并启动。测试连接会与代理模块进行验证。

新增

修改

删除

刷新列表

机器名

testmysql

地址

10.16.16.17

☒ 启用

产品类型

mysql

描述信息

连接方式

☒ 直接连接

监听端口

3306

数据库名

mydb

URL

jdbc:mysql://\$host:\$port/\$database

用户名

sa

口令

●●●●●●

☐ 通过代理

代理端口

测试连接

数据库定义列表(可搜索)

名称	机器地址	产品ID	监听端口	数据库名	URL	用户	代理端口	启用	描述信息
...	10.16.16.17	oracle 11g	1521	...	jdbc:oracle:thin@...	...	0	停用	...
...	10.16.16.31	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	启用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	启用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	启用	...
...	10.16.16.9	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	启用	...
...	10.16.16.5	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
...	10.16.16.8	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	启用	...
...	10.16.16.1	oracle 11g	1521	...	jdbc:oracle:thin@\$host:\$port/\$servicename	...	0	停用	...
testmysql	10.16.16.17	mysql	3306	mydb	jdbc:mysql://\$host:\$port/\$database	sa	0	启用	

4.1.4) 配置 -> 主机

完成对需要监控的数据库的增、删、改操作。

- 连接类型，支持通过 ssh , telnet 连接主机。
- 提示符，是指定的连接用户登录主机后的命令行提示符(P\$1 变量)

主机 ×

新增 修改 删除 刷新列表

主机名: DNS服务器 地址: 10.12.65.73 ☒ 启用

产品类型: RHEL 5 描述信息:

连接方式: ☒ 直接连接 类型: SSH 端口: 22 提示符PS: #

用户名: root 口令: ●●●●●●●●●●

☐ 通过代理 代理端口: 测试连接

主机定义列表 (可搜索)

主机名	机器地址	产品ID	连接方式	端口	用户	提示符PS	代理端口	启用	描述信息
DNS服务器	10.12.65.73	RHEL 5	SSH	22	root	#		<input checked="" type="checkbox"/> 启用	

4.1.5) 配置 -> 应用服务

完成对需要监控的应用服务的增、删、改操作。

- 连接方式，支持 http, jmx, snmp 3 种方式。
http 方式，访问端口，获取 http 返回值及页面内容
snmp 方式，获取 jmx bean 的值
snmp 方式，通过 snmp 端口，获取 oid 的返回信息

应用服务 ×

新增 修改 删除 刷新列表

主机名: 生产应用 (10.12.66.10) 地址: 10.12.66.10 ☒ 启用

产品类型: weblogic 描述信息: web服务器

连接方式: HTTP 测试连接

HTTP 端口: 7001

JMX

SNMP

主机定义列表 (可搜索)

主机名	机器地址	产品ID	连接方式	端口	用户	提示符PS	代理端口	启用	描述信息
生产应用 (10.12.66.1)	10.12.66.1	weblogic	HTTP	7001				<input checked="" type="checkbox"/> 启用	web服务器
生产应用 (10.12.66.10)	10.12.66.10	weblogic	HTTP	7001				<input checked="" type="checkbox"/> 启用	web服务器

4.1.6) 配置 -> 网络设备

完成对需要监控的网络设备的增、删、改操作。

- 类型，支持 snmp 协议访问设备
- 协议，有 UPD, TCP 两种
- 端口，UPD 一般对应端口 161, TCP 对应 162
- 读权限，是 SNMP 协议访问设备时的团体字
- 写权限，目前不支持对网络设备的写操作

网络设备 ×

设备名
 地址
☐ 启用

产品类型
 描述信息

连接方式
 类型
 端口
 协议

读权限
 写权限

网络设备定义列表

设备名	机器地址	产品ID	连接方式	端口	用户	提示符PS
1.1.0.98	1.1.0.98	cisco	SNMP	161	public	UDP
1.1.0.98	1.1.0.98	cisco	SNMP	161	public	UDP
1.1.0.98	1.1.0.98	cisco	SNMP	161	public	UDP
h3c1	10.16.16.254	h3c v11	SNMP	161	public	UDP

4.1.7) 配置 -> 管理组

管理组是把同类别的监控源分组。在管理列表树中显示为一个目录。一个监控源可以被分到多个组中。

- 类别： 可选数据库、主机、网络设备、应用服务， 分组只能在同一类别监控源中选择
- 执行方式 批量执行时的顺序
- 并发度 可以多并发进行批量执行， 无并发则逐一在多个监控源上执行脚本

管理组 ×

组名称
 类别
 分组条件

执行方式 ☒ 无序执行
 ☐ 主节点后执行
 ☐ 主节点先执行
 ☐ 按升序执行
 ☐ 按降序执行
 并发度

描述信息

组定义列表

组名	类别	分组条件	执行方式	描述...
地市下行	DB		无序执行	
地市主机	OS		无序执行	
地市应用	DB		无序执行	

组成员列表

机器名	组名	类别	分组条件语句	角色
杭州	地市下行	DB		普通成员
杭州-县区	地市下行	DB		普通成员
杭州-应用	地市下行	DB		普通成员

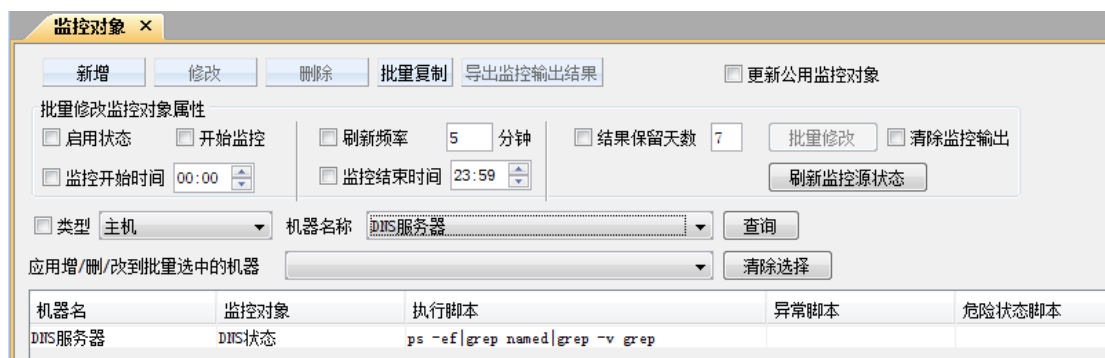
- 管理组成员 选择组中包含的组员，调整组员在列表中的顺序
 左边列表是可选择的机器， 右边是已经包含的成员
 ->可以把选中的机器添加到组 <- 是相反操作，把右边选中的机器从组中剔除
 =>把左边所有机器添加到组 <=是相反操作，移除所有组成员
 ↑ ↓ 调整成员顺序
 分组条件未使用



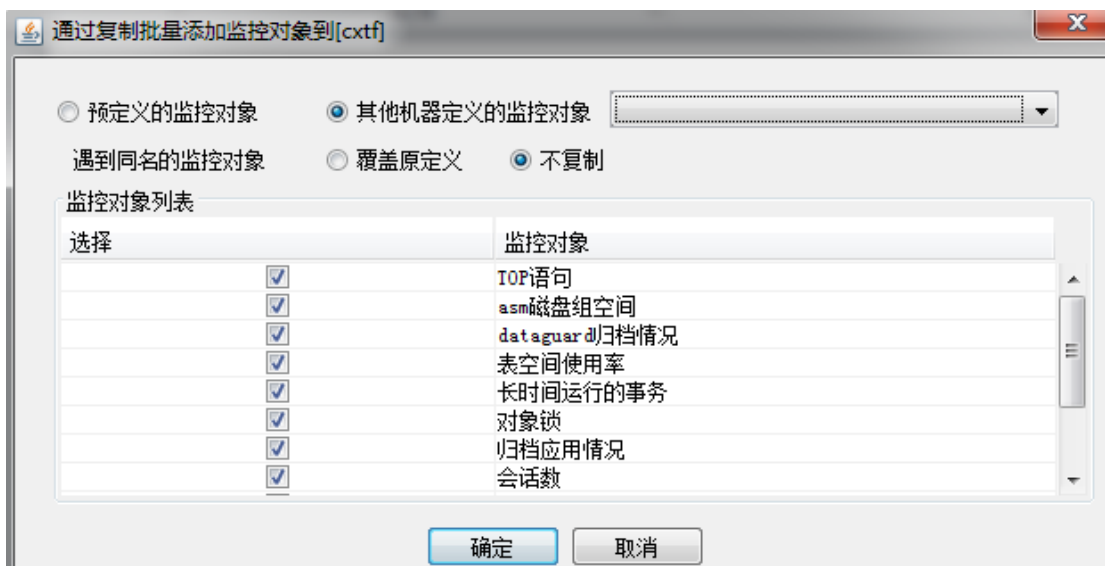
4.1.8) 配置 -> 监控对象

监控对象界面定义监控调度模块要采集的信息。监控对象有预定义的公用对象，也可以自由定义。

- 新增 选择某个机器后，可以添加新的监控对象，弹出新窗口进行定义
- 修改 在列表中选中已经定义的监控对象，弹出新窗口进行修改
- 删除 删除已有定义，同时清除以前产生的监控输出结果，可能耗时久，一般采用停用方式。



- 批量复制 把其他监控源定义的监控对象或预定义的监控对象批量复制到选中的机器。
遇到同名监控对象可以选择两张处理方式：
覆盖原定义 替换原有的监控对象定义
不复制 不进行复制



- 开始监控标记 选择开始监控，否则停止收集监控信息
- 监控类型 选择作为监控采集命令，否则只在客户端中使用
- 作为定制命令执行 出现在“管理列表”右键菜单的定制命令里。
- 发送告警消息 是否通过消息发送模块发送告警信息
- 列分割符 主要用在输出结果是文本时，区分列，默认是空格(tab)，特殊字符需要转义\，如 \|，以|为分隔。
- 结果集模式 单行： 对多行的输出结果，按项目进行 count, max, min, avg 等汇总；
多行： 注意标识结果的唯一性，可以通过期望行数进行告警评估
- 刷新频率 监控调度模块定时采集的时间间隔。最小值为 2 分钟。
- 结果保留天数 输出结果保留在历史表中的时间。超过时间会自动清理
- 有效行开始定义，有效行结束定义 适用于文本输出结果，选择有效的结果进行评估。可选项：行数， 有效的匹配字符串，开始于匹配的字符串。
- 监控开始时间、结束时间 定义监控调度采集的时间范围
- 执行脚本 定义监控对象执行的脚本。主机 – 执行命令或 shell 脚本； 数据库 – 执行 sql 脚本； http 连接 – 执行 <http://host:port/xx> ,访问页面； SNMP 方式 -- 填写 OID （可通过管理工具 -> SNMP-MIB 查询界面进行查询）； JMX 方式 – 填写 MBEAN(可通过管理工具 -> JMX-MBEAN 查询界面进行查询)。
- 测试执行按钮 测试执行脚本，进行结果评估等。测试结果可查看输出窗口、命令行的输出窗口
- 表达式帮助按钮 查看表达式基本帮助信息
- 监控对象上的正常结果表达式、异常结果表达式、危险结果表达式，分别定义了对监控结果根据表达式进行评估，是否产生预警。表达式定义语言使用的是开源的 IExpression，可参看 config 目录下的相关文档。表达式评估顺序按照危险 -> 异常 -> 正常，遇到评估结果为 TRUE 的不继续进行评估。
- 项目属性 监控对象可以定义子项目，获取更明细内容。如果有定义“多行期望值”，必须定义至少一个项目。
“项目名称”定义可标志或易读的名字。预定义变量 NAME__ ,取该项目的实际值，
LINE_CONTENT 变量，包含整行内容。
对于 HTTP 访问方式，预定义变量 CODE 表示返回代码, OK 为正常， 其他是数字；
预定义变量 CONTENT 是页面内容。这些可以用在表达式脚本中。
“列名/列数”和“名称对应的列名/列数”填写方式一致；
表达式定义和“监控对象上的表达式”填写方式一致。区别在于，该处对每行记录的每个项目分别评估，“监控对象上的表达式”只做一次评估；
结果集函数 用于在把多行结果合并成单行时使用。有 count, max, min, avg, sum 几种。
- 项目列表 列出监控对象包含的项目。在进行修改项目、删除项目时，需要先点击列表中的一个，进行操作。

4.1.9) 配置 -> 产品信息

产品信息，定义监控源对应的产品类型。主要用途在 jdbc 连接时使用的 URL 串。
选择类型， 可选输入“厂商”， “产品名称”， 缺省驱动 URL。

产品信息 ×

保存 删除

类型 数据库 厂商 oracle 产品名称 oracle db

产品ID* oracle 11g 缺省驱动URL jdbc:oracle:thin:@\$host:\$port/\$servicename

产品信息列表(可搜索)

产品ID	类型	厂商	产品名称	缺省驱动URL
AIX 5L	OS	ibm	AIX	
AIX 6L	OS	ibm	AIX	
AIX 7L	OS	ibm	AIX	
ase15	DB	sybase	sybase ase	jdbc:sybase:Tds:\$host:\$port/\$database
cisco	NET	cisco	cisco	
db2 v9	DB	ibm	db2	jdbc:db2://\$host:\$port/\$database
h3c v11	NET	h3c	h3c	
hp-ux	OS	hp	hp-ux	
Lotus Notes	WEB	ibm	lotus notes	
mssql	DB	microsoft	sqlserver	jdbc:sqlserver://\$host:\$port/\$database
mysql	DB	oracle	mysql	jdbc:mysql://\$host:\$port/\$database
oracle 10g	DB	oracle	oracle db	jdbc:oracle:thin:@\$host:\$port/\$servicename
oracle 11g	DB	oracle	oracle db	jdbc:oracle:thin:@\$host:\$port/\$servicename
oracle 8i	DB	oracle	oracle db	jdbc:oracle:thin:@\$host:\$port/\$servicename
oracle 9i	DB	oracle	oracle db	jdbc:oracle:thin:@\$host:\$port/\$servicename
RHEL 5	OS	redhat	LINUX	
solaris	OS	sun	solaris	
tomcat	WEB	apache	tomcat	
weblogic	WEB	oracle	weblogic	
websphere	WEB	ibm	websphere	
windows	OS	microsoft	windows	

4.1.10) 配置 -> 程序选项

程序选项配置数据库连接信息，以及监控调度模块的配置。


- 主数据库配置

配置 m4db 系统使用的数据库。目前内置了 oracle, mysql, derby 的驱动。

选择“驱动类”， URL 自动填充，修改其中的变量值；

填写“用户名”，“口令”， 可以点击“测试连接”，测试可否连接该数据库。

- 服务器信息

【可选填】定义通过客户端“监控 -> 监控结果展示”，或者 ，使用本地浏览器打开监控展示 WEB 页。

- 监控程序配置

定义监控调度模块使用的端口，用于客户端发送一些指令：检查监控调度程序状态，清理缓存的监控对象，停止监控等。

刷新间隔：定义调度程序多久执行一次，一般是 2 分钟。

间隔超时：一次调度超过多久，终止该线程，重新开始新线程。

程序选项 ×

保存配置

主数据库配置

驱动类 oracle.jdbc.driver.OracleDriver

URL jdbc:oracle:thin:@144.16.16.97:1521:m4db

用户名 mydb 口令 ●●●●●● 测试连接

服务器信息

Web服务器地址 127.0.0.1 端口 8200 ☐ 和数据库地址一致

浏览器位置 s (x86)\Mozilla Firefox\firefox.exe 测试打开

许可信息

注册码 90e5bbe15bbe0be13a56332f1f9f9ac127a39f20eefdae6

监控程序配置

监听端口 7511

监控程序运行的地址 144.16.16.97 ☐ 和Web服务器地址一致

刷新间隔(>2分钟) 2 间隔超时(分钟) 10

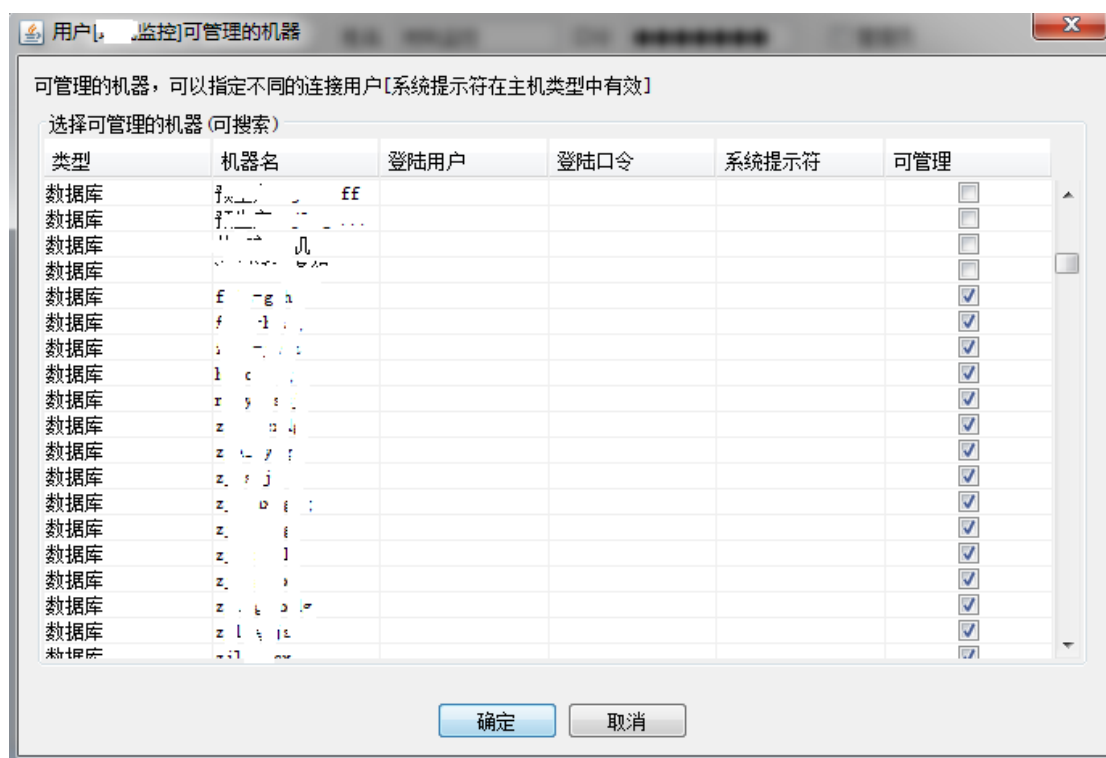
4.1.11) 配置 -> 用户管理

新增用户，可以登录“监控结果展示”页面，”客户端“软件的用户，并定义该用户可以查看到的机器。

管理员：可以查看所有机器。



“可管理的机器”，定义该用户能查看及执行操作的机器。可以改变用户使用不同的登录用户连接被管理的机器。



4.1.12) 配置 -> 机器扩展属性

“机器扩展属性“，可以给机器添加一些信息，在监控展示页面查看监控源属性时获取更多的信息。

机器扩展属性 x

类型: 主机 机器名称:

从其他机器复制: ☐ 覆盖同名属性 ☒ 跳过同名属性 复制

扩展属性

属性名: 责任人

属性值: 测试ABC

加入到列表 更新到列表 从列表删除 上移 下移 保存 全部清除

属性名	属性值
机房位置	P1-L1
责任人	测试ABC

查看监控源信息

名称: DNS服务器 地址: ...

产品: RHEL 5 连接使用的端口: 22

连接用户: root 连接方式: SSH

提示符PS: #

描述信息: -

扩展属性:


属性	内容
机房位置	P1-L1
责任人	测试ABC

4.1.13) 配置 -> 刷新管理列表

当新增了监控源，定义了“管理组”后，可以通过该功能重载管理列表。
也可以直接在“管理列表”窗口中右键菜单进行刷新。

4.1.14) 管理工具 -> 新脚本

打开一个新的“脚本”窗口，用于输入脚本。脚本统一是 sql 语言方式高亮的。

也可以通过快捷按钮  方式打开。

4.1.15) 管理工具 -> 运行命令

执行当前脚本窗口中的命令。

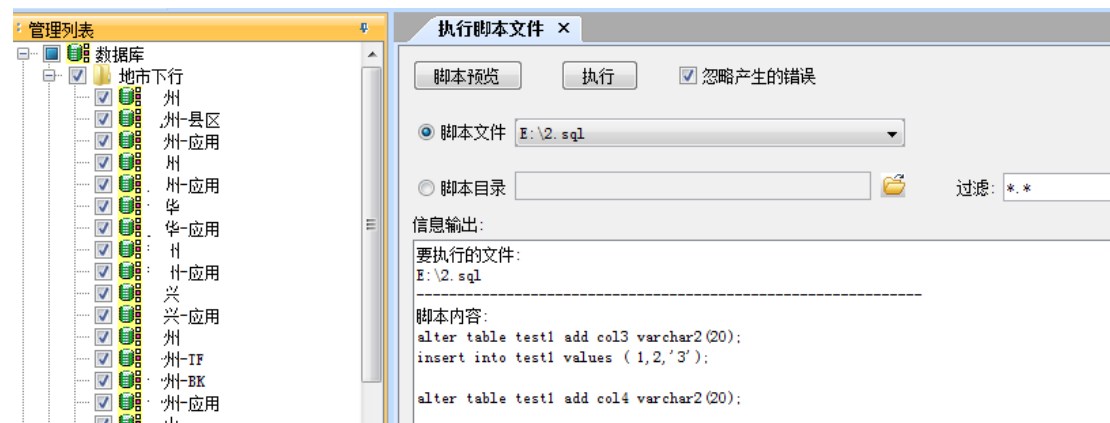
不同的监控源类型，执行命令的方式不同。根据监控源定义时使用的连接，发起命令。
其中以“代理方式连接”的，命令发送到被监控的机器上，通过调用 java 的 system 命令方式，返回输出结果。

JMX , SNMP 的执行, 提供了工具: 管理工具 -> SNMP-MIB 查询 , JMX-MBEAN 查询。

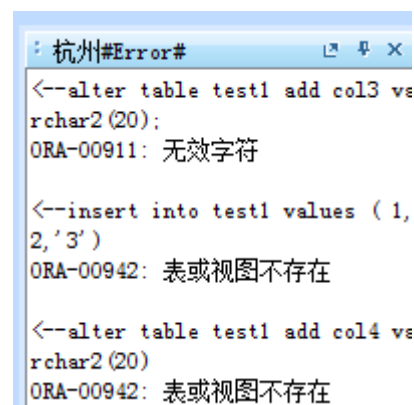
4.1.16) 管理工具 -> 执行脚本文件

“执行脚本文件”, 以并发、批量方式在多个同类型监控源上执行脚本文件包含的命令。也可以执行某个目录下的所有文件(“过滤”匹配文件名)。

“脚本预览”, 可以查看要执行的脚本内容, 已经按目录执行时读取文件的顺序。



可以选择是否“忽略产生的错误”。忽略错误, 在逐条执行语句过程, 跳过发送错误的命令, 继续执行, 并将发送错误的语句记下。执行完成后产生“监控源名称#Error#”的 TAB 窗口。



4.1.17) 管理工具 -> 输出结果比较/导出

“输出结果比较/导出”, 对输出窗口产生的文本内容, 或者表格内容进行两两比较, 标记出差异。

“左输出内容”勾选后, 从输出窗口中选取一个窗口的内容。如果不勾选, 则左边的内容可以进行编辑、粘贴操作。

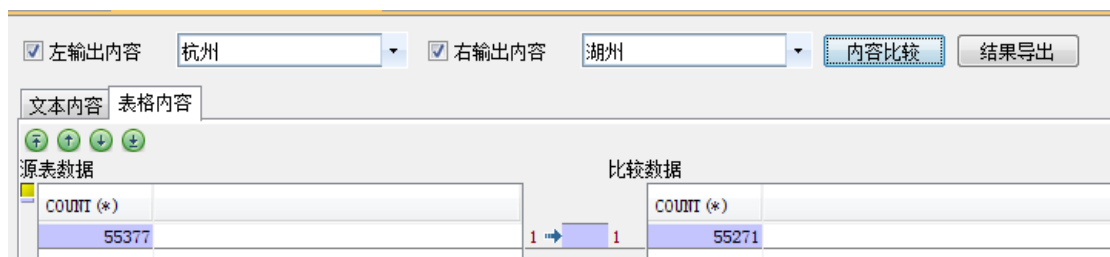
“右输出内容”操作同左。

点击“内容比较”, 不同的内容会被标记出来。

点击“结果导出”, 会把所有打开的窗口内容导出到文件中。



“表格内容“操作方式通”文本内容“比较。但选择的输出内容是表格形式的窗口。如：

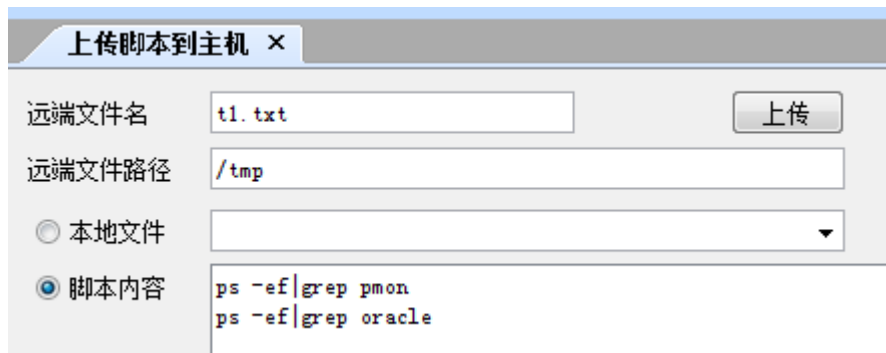


4.1.18) 管理工具 -> 关闭输出窗口

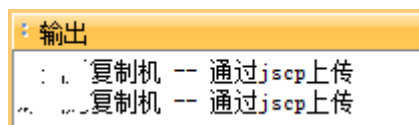
该功能不打开新的窗口，可以批量的关闭所有在执行过程中产生的结果输出窗口，错误信息窗口。

4.1.19) 管理工具 -> 上传脚本到主机

该功能可以把本地的文件或脚本内容，批量传送到多个主机上。传输到主机的文件，目录、名称一致。



可以在“脚本内容“处输出文件包含的内容，上传到指定的目录，并命名为指定文件。也可以选择“本地文件“。传输结果反应在主输出窗口中。

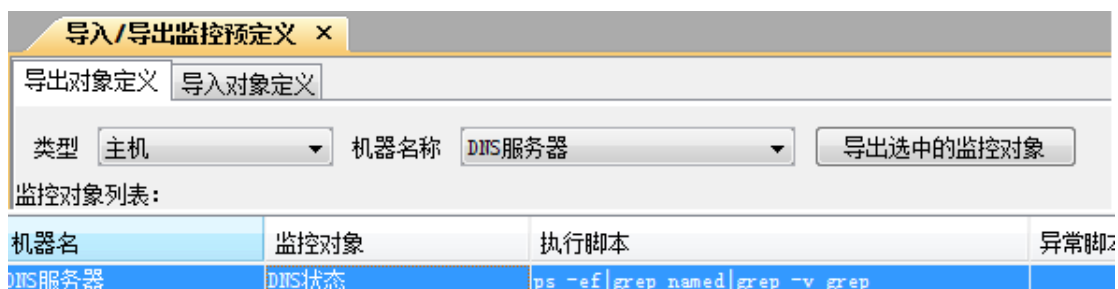


4.1.20) 管理工具 -> 导入/导出监控预定义

可以通过导入、导出监控对象的定义，实现分享。导出内容包含监控对象定义、产品定义信息等。

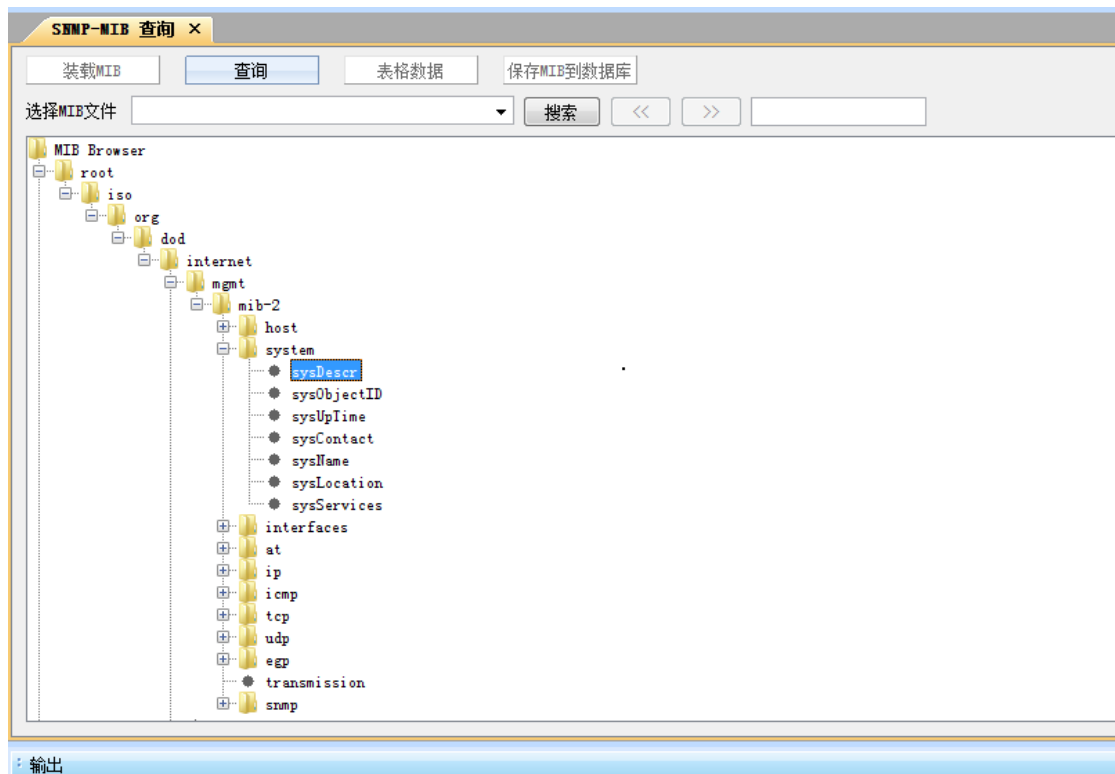
导出是从普通的监控源定义的对象中选择监控对象。

导入到预定义监控对象中，可以在监控对象定义中“从公用对象列表获取“列表中选取。



4.1.21) 管理工具 -> SNMP-MIB 查询

SNMP-MIB 查询提供工具，可以在 MIB 列表选取要查询的 OID，查询结果。



输出

语法 : DisplayString (SIZE (0..255))

状态 : mandatory

描述 :
"A textual description of the entity. This value should include the full name and version identification of the system's hardware type, and networking software. It is mandatory that this only contain printable ASCII characters."

类型 : 普通记录

打开页面会列出装载的 MIB 表:

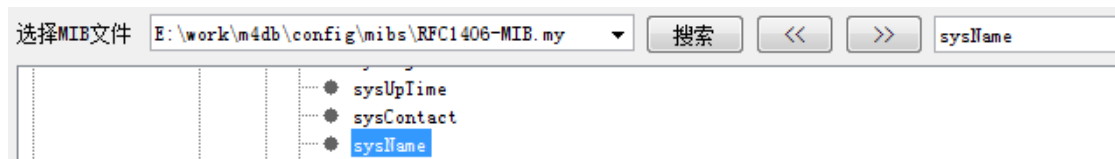
输出

装载的MIB:[CORE-MIB, IOS-LIC, IOS-NEISIAIE, IOS-SMI, CISCO-MEMORY-POOL-MIB, CISCO-PROCESS-MIB, F5-BIGIP-APM-MIB, F5-BIGIP-COM-LOCAL-MIB, F5-BIGIP-SYSTEM-MIB, HOST-RESOURCES-MIB, RFC1213-MIB, IOS-IV, IOS-SYSSIAIE, IOS-SV, IOS-SYSINFO, IOS-UC]

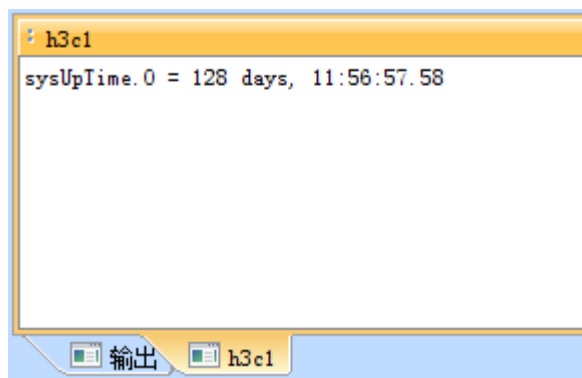
可以“选择 MIB 文件”，点击“装载 MIB”添加 MIB 到列表 MIB Browser，可以点击“保存 MIB 到数据库”到数据库中。

通过点击 MIB 树，查找要查询的信息，描述信息输出在主输出窗口中。

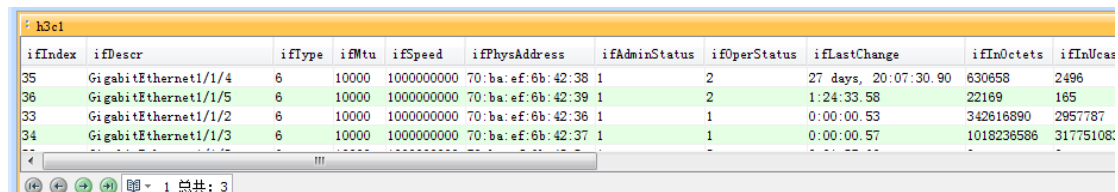
可以“输入搜索内容”，点击“搜索”，查找节点名称



普通记录点击“查询”，输出结果:



表格类型记录，点击“表格数据”，输出表格信息：

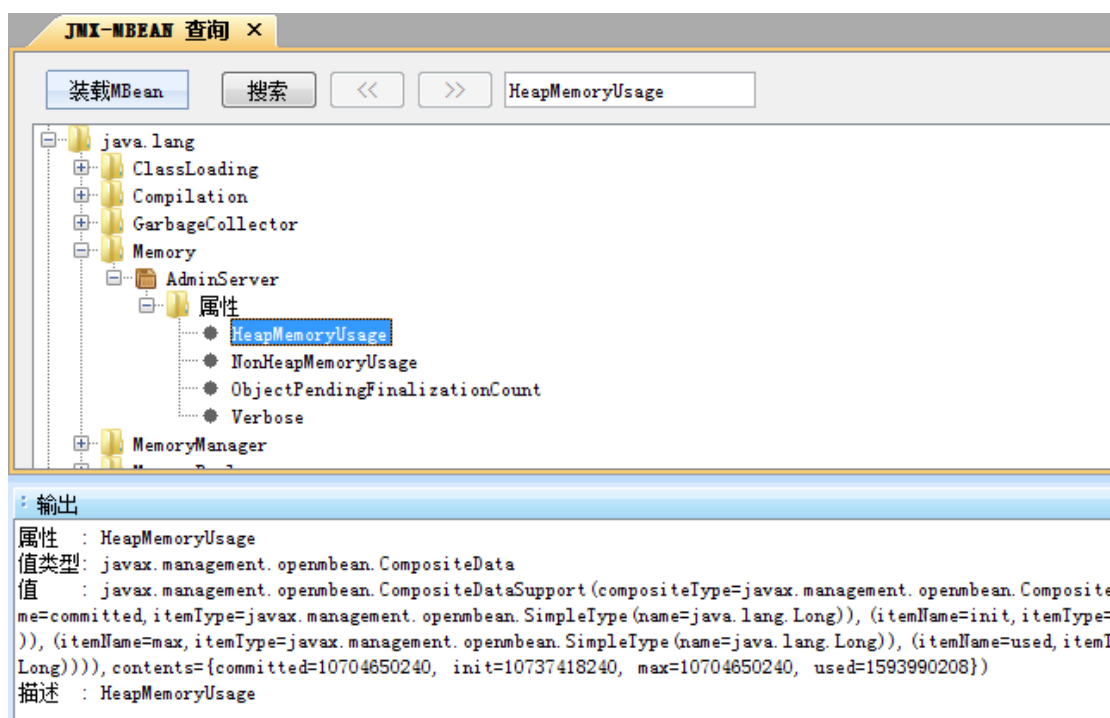


ifIndex	ifDescr	ifType	ifMTU	ifSpeed	ifPhysAddress	ifAdminStatus	ifOperStatus	ifLastChange	ifInOctets	ifInUcas
35	GigabitEthernet1/1/4	6	10000	10000000000	70:ba:ef:6b:42:38	1	2	27 days, 20:07:30.90	630658	2496
36	GigabitEthernet1/1/5	6	10000	10000000000	70:ba:ef:6b:42:39	1	2	1:24:33.58	22169	165
33	GigabitEthernet1/1/2	6	10000	10000000000	70:ba:ef:6b:42:36	1	1	0:00:00.53	342616890	2957787
34	GigabitEthernet1/1/3	6	10000	10000000000	70:ba:ef:6b:42:37	1	1	0:00:00.57	1018236586	31775108

4.1.22) 管理工具 -> JMX-MBEAN 查询

JMX-MBEAN 查询工具，通过 JMX 协议访问监控源，“装载 MBean”查询出所有的 MBEAN 信息。通过点击 MBEAN 树，或者“搜索”属性名称查找内容。

主输出窗口内容包含 Mbean 的属性、值类型、值、描述等信息。



4.1.23) 监控 -> 开始监控

无窗口命令，在监控调度模块已经启动的情况下，可以重启监控调度。

4.1.24) 监控 -> 停止监控

无窗口命令，在监控调度模块已经启动的情况下，可以停止监控调度。

4.1.25) 监控 -> 检查监控状态

无窗口命令，在监控调度模块已经启动的情况下，查询监控调度进程的执行情况。输出到主

输出窗口：



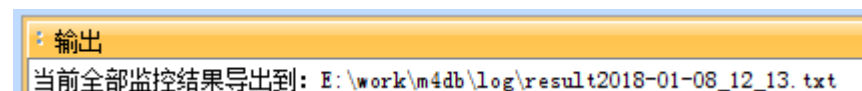
监控进程启动时间，上次执行调度的时间，监控的进程 ID，总执行次数，以及当前在执行的监控对象信息。

4.1.26) 监控 -> 清除缓存对象

无窗口命令，在监控源、监控对象发生变化后，将变化后的结果重新装载到监控调度中。

4.1.27) 监控 -> 导出全部监控结果

无窗口命令，将监控对象的当前结果输出到文本文件中。



4.1.28) 监控 -> 监控结果展示

通过“程序选项”中定义的浏览器，打开监控结果展示页面。

4.1.29) 帮助

帮助文件位于 config 目录下，包含简单的帮助内容，用户手册。

4.2. 监控结果展示

监控结果展示是 web 界面展现监控调度模块的输出，提供对告警的处理，历史数据查询分析，拓扑图的定义等。使用的登录用户和客户端软件一致。

通过浏览器打开应用服务如：<http://127.0.0.1:8200/m4web>，使用“用户管理”定义的用户口令登录系统。

登录用户: admin 登录时间: 2018/01/08 12:50:32 退出登录

浙江

点击“登录用户”，可以进行口令修改。

4.2.1 实时监控

实时监控页面展示监控调度模块的输出结果。见上图。

展示分 4 种方式：

■ 实时看板

监控系统整体概况：监控对象按主机、数据库、应用服务、网络设备分别显示各种状态的数值；告警处理情况：总监控的主机、数据库等数量。

2 个动态性能图，包含 CPU%，IO 读写情况，内存使用%。



■ 业务视图

定义业务逻辑关联的拓扑信息。每个节点可以包含多个监控源。节点间可以连线，添加说明。

节点的状态信息是包含的监控对象的评估结果汇总。按告警严重程度显示状态，分危险、警告、未知、正常 4 种；对应颜色为红色、黄色、灰色、绿色。节点名称后面的数字，红色表示发生告警的监控对象数量，黑色表示表示告警处理的数量。

如果告警处理数量和告警数一样，状态显示恢复为正常。



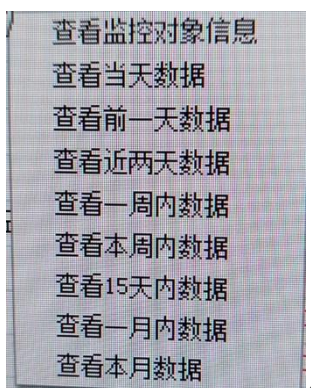
正在处理中: 1/2

表示有两个告警对象，其中一个在处理。

点击节点，展示该节点包含的所有监控源的监控对象输出。

实时监控							
监控结果查看(其他应用)							
类型	监控源	监控对象	执行时间	监控结果	耗时 (s)	上次结果	上次时间 (结果变化)
主机	192.168.1.100	硬盘信息	2018/01/08 14:16:24	正常	0.77	正常	2018/01/07 19:46:20
主机	192.168.1.100	硬盘信息	2018/01/08 14:06:11	警告	1.85	正常	2018/01/04 00:30:20
主机	192.168.1.100	网络状态	2018/01/08 14:14:12	警告	1.95	正常	2017/12/29 09:32:22
主机	192.168.1.100	网络状态	2018/01/08 14:14:12	正常	1.97	正常	2017/09/26 12:31:57

点击某个“监控源”，可以查看监控源的基本信息，以及近期历史数据：



监控源的基本信息包括名称，IP 地址，产品类型，端口，用户，连接方式，扩展属性等：

查看监控源信息

名称: NTP服务器89	地址: 89
产品: RHEL 5	连接使用的端口: 22
连接用户: jmonitor	连接方式: SSH
提示符PS: ~]\$	
描述信息:	null
扩展属性:	

属性	内容

多个时间范围的历史数据，具体见“历史数据”部分说明。

点击某个“监控对象”，查看监控对象的基本信息，刷新时间，脚本等：

查看监控对象信息

监控源: ' '	监控对象: OGG复制状态
刷新时间(分): 20	结果保留天数: 190

执行脚本:

```
/goldengate/ggs/ggscinfo.cmd
```

点击某个“监控结果”，查看监控输出的明细内容，包括监控项目输出、评估结果，脚本执行内容原始输出等。

监控结果明细
goldengate

处理人:

预计处理完成时间: 2018/01/08 14:40:18

异常处理

情况说明: 处理

监控项目明细:

☒ 列表
 ☐ 横向表

导出Excel

项目	结果	值
RJ T L Chkpt延迟		00:05:58
RJ T L 延迟		00:00:00
RJ T LL 状态		ABENDED
E S T Chkpt延迟		00:00:08
E S T 延迟		00:00:00
E S T 状态		RUNNING
E A T Chkpt延迟		00:00:01
E A T 延迟		00:00:00
I M 状态		RUNNING

原始输出:

2018-01-08 14:20:15.959524 RBA 255686066

REPLICAT F

Last Started 2017-05-19 21:58

Status RUNNING

Checkpoint Lag 00:00:00 (updated 00:00:01 ago)

Log Read Checkpoint File dirdat/gshx/tz000749

2018-01-08 14:20:13.996013 RBA 147211439



结果

异常情况处理

值

×

预警发生时间

2018/01/08 14:22:26

预计处理时间

0

↑

↓

小时

15

↑

↓

分

☒ 期间消除预警

预警处理范围

☒ 本监控对象 ☐ 本监控源 ☐ 本机器(同一IP)所有监控

情况说明

开始异常处理

取消

主要是标记该告警已经知晓，并开始进行处理。

“情况说明”内容，显示在列表的”备注“信息内，便于查看处理情况。

告警处理范围可选“本监控对象“，”本监控源“ - 同一机器名称，”同一 IP 的所有监控源“。

■ 表格显示

按告警严重程度由高到低，展示监控对象输出。

“备注“列显示了在处理的告警情况说明，点击，可以查看该监控对象所有发送的告警处理历史情况。

类型	监控源	监控对象	执行时间	监控结果	耗时(s)	上次结果	上次时间(结果变化)	备注
	1	ASM可用空间	2018/01/08 14:06:10		0.28		2017/12/05 14:12:16	在处理...情况说明...
	1	SSD状态	2018/01/08 14:06:11		1.85		2018/01/04 00:30:20	
	1	城市最复制状态	2018/01/08 13:54:18		3.74		2018/01/08 10:54:21	
	1	城市最复制状态	2018/01/08 13:56:14		1.10		2018/01/04 09:12:34	
	1	文件系统空间	2018/01/08 13:54:17		0.81		2017/11/03 07:06:01	
	1	文件系统空间	2018/01/08 14:10:11		1.57		2017/09/26 12:31:57	

列表包含预警发生时间、处理人、情况说明、完成时间、处理结果、状态等。

预警状态可以手工在“预警及处理“页面中消除，也可以由监控调度模块定时更新：逾期消除、逾期未完成等。

预警发生时间	处理人	情况说明	完成时间	处理结果	状态
2018/01/05 16:34:11	-	观察	--		逾期未完成
2017/09/26 08:51:34	-	处理	--	状态恢复正常	逾期消除
2017/08/28 09:23:32	-	新加的队列，添加了dirchk文件权限	--	状态恢复正常	逾期消除
2017/07/21 06:43:37	-	一体机查询使用，删除制队列	--	状态恢复正常	逾期消除
2017/07/17 08:03:37	-	一体机重启，启动hogg	--	状态恢复正常	逾期消除
2017/07/17 07:23:37	-	一体机重启，启动hogg	--		逾期未完成
2017/06/12 16:43:21	-	已处理	--	状态恢复正常	逾期消除
2017/05/04 13:58:57	-		--		逾期未完成
2017/05/04 13:37:00	-		--		逾期未完成
2017/05/04 08:28:57	-		--		逾期未完成

点击“表格显示“，可以对监控源，监控对象进行过滤。

填写内容后“确定“，匹配名称进行匹配。

“清空“，清除过滤条件，全显示。

☒ 表格显示 ☐ 机房视图 刷新时间

过滤条件 监控对象


监控源

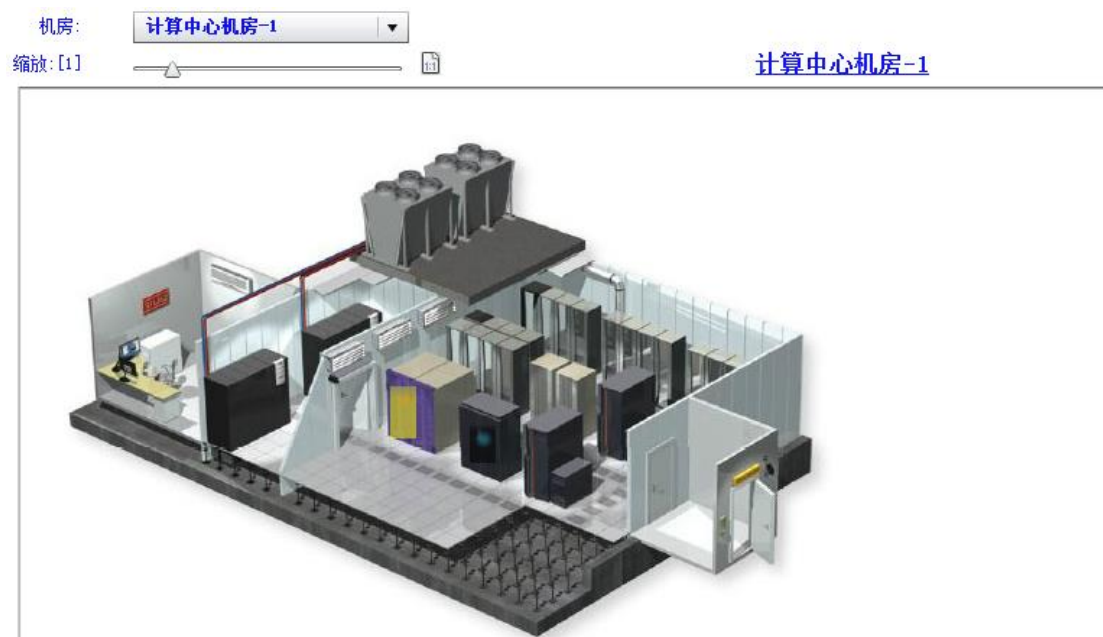
监控对象

清空 确定 取消

■ 机房视图

通过装载图片，显示机房的物理位置信息。

点击“机房视图”，之前选择的“机房”会自动载入。图片的显示比例为 1:1，可以拖动“缩放”标尺，调节大小。点击，恢复到 1:1 的大小。



双击机房上定义的锚点，可以打开机柜视图。



双击机柜上定义的锚点，查看与该锚点关联的监控源定义的所有监控对象输出内容。

类型	监控源	监控对象	执行时间	监控结果	耗时(s)	上次结果	上次时间(结果变化)	备注
生产库		表空间使用(不可扩展)	2015/04/10 11:37:25	警告	1.28	警告	2014/11/21 23:24:35	
生产库		dataguard归档	2015/04/10 09:30:00	正常	0.00	正常	2015/04/10 09:30:00	
生产主机		vmstat	2015/04/10 09:30:00	正常	0.00	正常	2015/04/10 09:30:00	
生产主机		vmstat	2015/04/10 09:30:00	正常	0.00	正常	2015/04/10 09:30:00	
生产库		涉及备库	2015/04/10 09:30:00	正常	0.00	正常	2015/04/10 09:30:00	
生产库		网络sql	2015/04/10 09:30:00	正常	0.00	正常	2015/04/10 09:30:00	

“刷新时间”是定时获取最新的监控结果的间隔时间。一般选择 2 分钟，和监控调度程序间隔保持一致。也可以点击”刷新“按钮，手工获取监控结果。

“上次刷新时间”，是本地发起刷新的时间；

“最近监控时间”，是监控调度主机上次执行调度的时间；

这两个时间会不断切换显示。

如果“最近监控时间”以红色显示，可能是本机时间比监控调度机器时间大很多，或者是监控调度程序异常导致。可以使用客户端中的“监控 -> 检查监控状态”，进行检查；具体问题可以登陆监控调度主机，查看监控的日志输出（nohup.out 文件）。

“拓扑图”，可以选择定义的不同业务视图。

点击“拓扑名称”，可以以全屏展示监控。再次点击，或 ESC 键，退出全屏。

4.2.2 历史数据

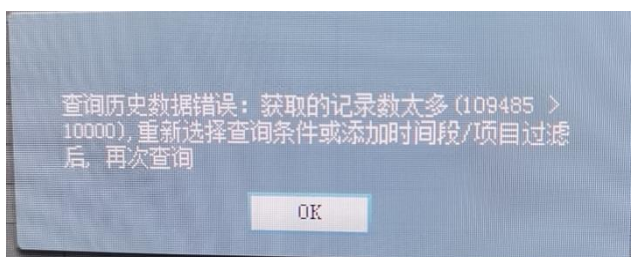
“历史数据”可以查询保留的监控结果输出，进行结果分析、图表展示。

可以选择“时间范围”，“监控源”，“监控对象”，进行查询。

“包含项目”，勾选，查询监控对象的子项目信息；不勾选，查询监控结果输出。

查询结果不易过大（<10000），否则浏览器内存可能溢出。

提示获取记录数太多时，可以通过再次选择时间段过滤/项目过滤等方式，减少记录数！



“相同对象”，查询当前最新的相同监控对象名称的输出结果。

查询出来的监控结果可以再次按照“时间段”，“项目过滤”，“内容筛选”等进行过滤。过滤后的结果“使用”图的方式进行展示。

☒ 时间范围

2018/01/01

2018/01/05

☒ 监控源

js-hzcg-a

☒ 监控对象

vmstat

☒ 包含项目

☐ 相同对象

查询

清除选项

导出Excel

☒ 时间段过滤

8:0-12:0,14:0-17

☒ 项目过滤

CPU空闲,CPU等待:

☐ 内容筛选

应用过滤

取消过滤











☐ 使用

折线图

刷新图

多图分析

结果排序

类型	执行时间	监控结果	监控源	监控对象	监控项目	输出结果
	2018/01/01 08:00:43		js-hzcg-a	vmstat	vmstat.CPU等待	5.33
	2018/01/01 08:00:43		js-hzcg-a	vmstat	vmstat.CPU空闲	75.66
	2018/01/01 08:04:28		js-hzcg-a	vmstat	vmstat.CPU等待	2 <div>记录数: 1076</div>
	2018/01/01 08:04:28		js-hzcg-a	vmstat	vmstat.CPU空闲	68.33
	2018/01/01 08:08:28		js-hzcg-a	vmstat	vmstat.CPU等待	2

勾选“时间段”过滤，主要用于关注某段时间内数据的变化（默认有两段工作时间）；

勾选“项目过滤”，仅展示特定的监控子项目；



项目过滤选择

ASM可用空间 [271chxzzg]

×

☒ 总空间

☒ 可用空间

☐ 计算项目的差值 (相邻时间的两条记录)

项目下不同子项处理方式:

汇总

▼

确定

取消

● 项目下不同子项处理方式（当一个项目有多个不同的名称时）：

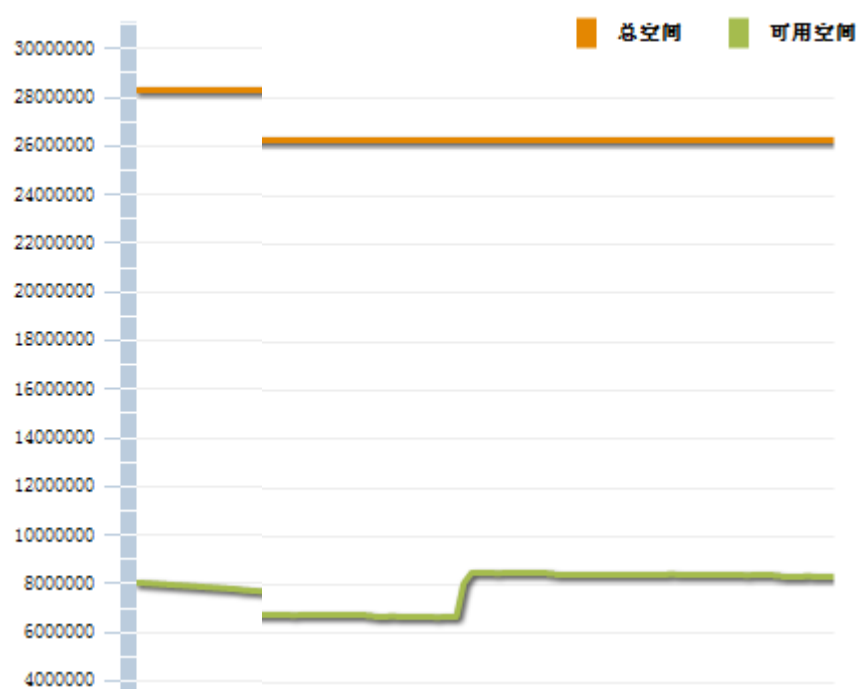
汇总（默认行为）、平均值、分别显示；

比如对 ASM 磁盘组的空间监控：

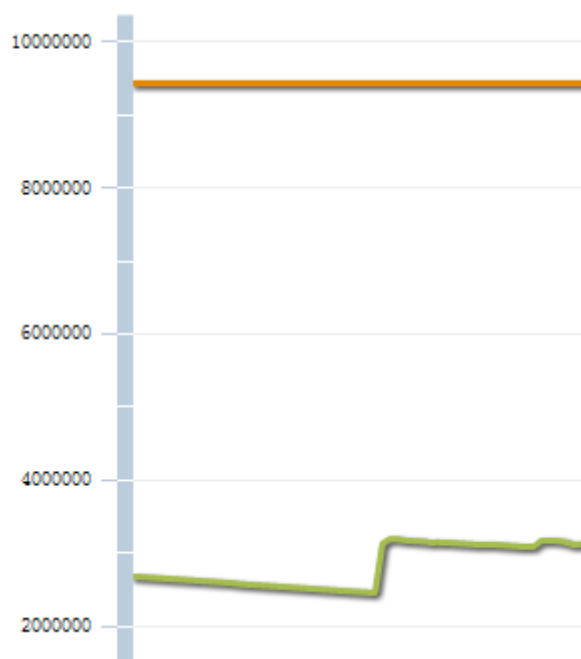
ASM可用空间	FRA. 总空间	614400
ASM可用空间	FRA. 可用空间	584430
ASM可用空间	ARCLOG. 可用空间	4158205
ASM可用空间	DATA. 可用空间	5603617
ASM可用空间	ARCLOG. 总空间	5120000
ASM可用空间	DATA. 总空间	22528000

包含 3 个 DG： FRA, DATA, ARCLOG 。

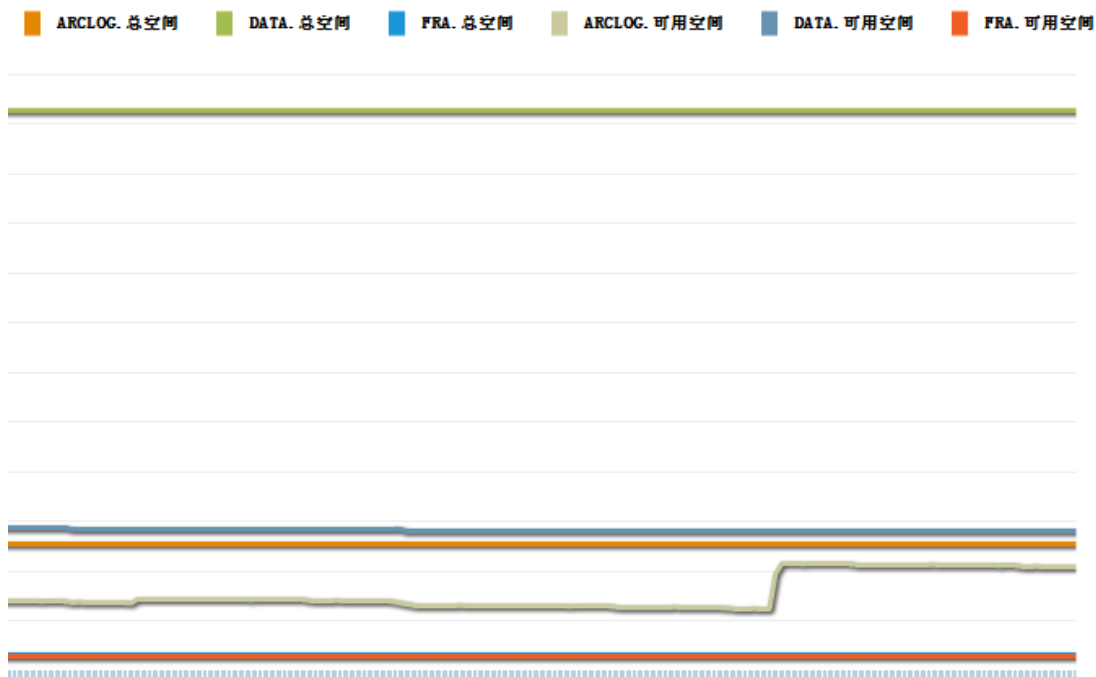
■ 在使用“汇总“方式显示图表时，会把 3 个 DG 的总空间、可用空间分别求和展示：



- 在使用“平均值”方式显示图表时，把 3 个 DG 的总空间、可用空间分别求平均值后展示：



- 而使用“分别显示”时，每个 DG 的总空间、可用空间都分别显示：

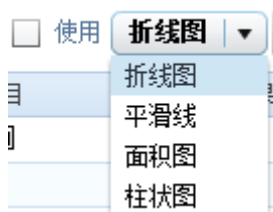


- 选项 ☒ 计算项目的差值 (相邻时间的两条记录) ，会在列表上显示当前项目值与上次的项目值的变动情况，可用很直观的查看数据的变化：

监控项目	输出结果	差值
ARCLOG. 可用空间	3139032	-24715

勾选“内容筛选”，可以过滤匹配监控项目名称；

点击“应用过滤”，根据选中的过滤条件对列表进行筛选，展示。

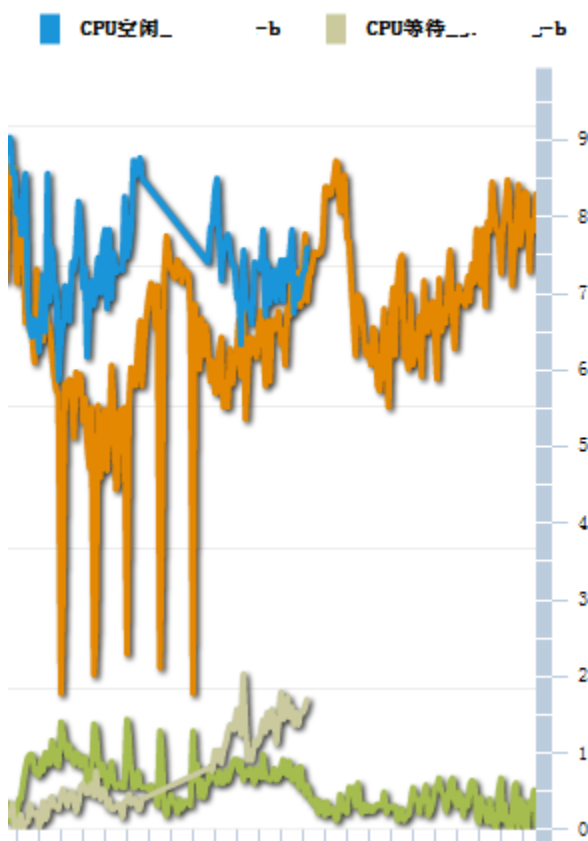


“使用图表”，提供了几种图形展示的方式，勾选后自动以图表方式展示。如果对过滤条件、图形类型进行了修改，点击“刷新图”，重新生成图表。

- “多图分析”功能主要用于对多个监控对象（可以包含不同的 Y 坐标，但 X 坐标时间序列要一致）的趋势分析融合在一张图表中，可以更加直观的进行关联分析。点击后弹出新的图表分析窗口，其中“时间范围”，“时间段过滤”不允许进行修改。生成图表后，添加到原图的方式有两种：
使用新的纵坐标系，主要用于两张图的项目值范围不一致，差异很大；
选择已有坐标系，从已经添加的图表中选取坐标系。
新的坐标系，坐标轴显示在右侧。



点击“添加到多图分析”，新的图表融合到主窗口的图表中。



4.2.3 预警及处理


“预警及处理”查询发生的告警，已经进行的“异常处理”，修改预警处理内容，结束预警填写解决方案。



可以选择“时间范围”，“监控源”，“监控对象”分别“查询预警发生”情况：

点击左侧“光标”图片，切换到操作模式。
双击节点，进行节点属性，包含的监控源设置。





“按名称搜索”，将名称匹配的监控源移到前面。
添加连线，先点击左侧的箭头线，在绘图区点击起始节点，移动鼠标，有线段出现，在终止节点点击，建立连线。双击连线，可以添加说明。

 **删除** 可以删除选中的节点，节点删除时，相关的连线一起删除。


 **撤销**  **重做** 对节点，连线的所有操作：建立、移动等可以回退，重做。

   **显示网格** 可以对齐多个节点，显示网格线。

 **保存** 将设计完成的拓扑图信息保存到数据库。包含节点及位置、大小，包含的监控源，连线等内容。

 **删除** 从数据库中删除该视图定义。

■ “机房视图”，上载机房图，添加“机柜”图到不同位置，“机柜”可以包含多个监控源。

点击  **新建机房**，输入名称，建立一个新的机房视图。



初始添加会弹出告警信息，提示需要上载机房背景图



如果新建的机房名称没有出现在左侧的列表中，可以点击 **刷新列表**。

添加或修改机房背景图，先在左侧列表中双击机房名称，右侧的绘图区域出现该机房的拓扑图信息。点击绘图区，可以选中背景图，进行拖动、改变大小等操作。

点击 **配置** 或者双击，上载背景图。



从 **装载背景图片** 弹出的文件选择窗口中选取图片文件，显示图片的原始大小，图片内容



确认图片正确，点击“确定”。图片显示到绘图区，机房的背景信息配置完成。

继续在机房视图的相应位置添加“机柜”信息。点击 [新建机柜](#)，输入机柜名称



同样操作装载机柜的图片。机柜的图片上可以添加锚点，与监控源关联。

点击 [锚点](#)，在机柜图片相应位置单击，或者鼠标按住左键并拖动，画出相应大小的方形。



锚点位置，大小可以通过鼠标操作进行修改。
双击绿色方形，将该锚点与监控源关联。



操作方式同“业务视图”中的节点设置。
所有设置完成的机柜列表，在左侧树中：

- ▼ 计算中心机房-1
 - APP-CUP1
 - YW2015

双击机房，或机柜，装载背景图片到绘图区，可以进行相应操作。

在机房背景图上，点击 **锚点**，将机柜与机房图上的位置进行对应。

点击按钮后在绘图区相应位置单击，或者鼠标按住左键并拖动，画出相应大小的方形。




，可以拖动修改位置，大小。双击该方形，将锚点与定义的机柜关联。



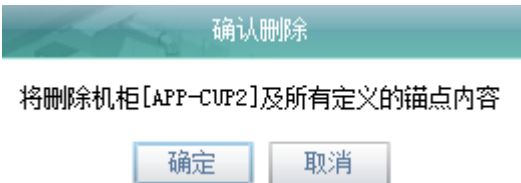
其他对图中锚点的操作，如 **删除**  **撤销**  **重做**，

        ☐ **显示网格** 同“业务视图”上的节点类似。

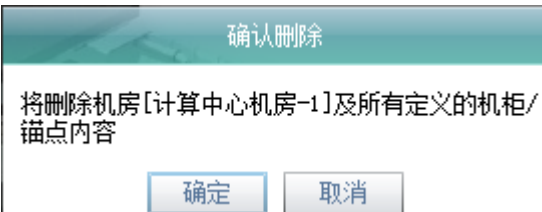
点击 **保存**，将当前定义的机房视图全部信息保存到数据库。

左侧的 **删除**，可以删除选中的机房或者机柜信息。

删除机柜时提示信息：



删除机房时，将所有机柜、锚点信息删除



4.3. 监控调度模块

“监控调度模块”定时发起对所有定义的监控对象进行数据采集、结果评估、保存。另有线程进行历史数据的清理、预警处理的逾期消除、检测没有定义监控对象的监控源是否可以连接等功能。监控程序读取 config/m4config.xml 中的 monitor 段定义,启动监听端口，用于接收客户端发起的指令。

Monitor.sh 是启动监控程序的脚本（最后加 -help，只打印出帮助信息）：

```
nohup java -Ddefault.client.encoding=GBK -Dfile.encoding=GBK -Dm4db.timeout.policy=only
-Dm4db.thread.policy=by-ip -Duser.language=zh -Dm4db.home=.. -jar lib/m4monitor.jar &
```

-Ddefault.client.encoding , -Dfile.encoding 指定客户端、产生文件的编码, 是 JAVA 的变量选项。

M4db.timeout.policy 是调度进程进行超时检测的策略, 有两个选项:

Only – 所有监控线程使用一个超时检测线程

Many – 每个监控线程使用自己的超时检测线程

M4db.thread.policy 是调度进程发起监控线程的策略, 有 4 种选项:

By-cat – 按监控大类 (主机、数据库、应用、网络) 启动 4 个线程

By-ip – 按定义的监控源分 IP 地址启动多个线程

By-host – 按监控源的名称启动多个线程

By-obj 按监控对象启动多个线程

M4db.ping.policy [false/true] , 采集前检查 IP 的连通性

M4db.syslog.threads , 接收 syslog 的线程数

M4db.syslog.port 接收 syslog 的端口(514)

M4db.syslog.maxrows 一次接收到最大数量的 syslog 后写入库

Snmp4j.listenAddress [upd:0.0.0.0/162] 接收 snmp trap 的监听 url

Snmp4j.tranThreads 接收 snmp trace 的线程数量

M4db.home 是 m4db 系统安装的路径

m4db.thread.cleanup 是否启动清理历史数据的线程, 默认启动。=false 则不启动, 可以通过手工清理或一直保留历史数据。

调度过程产生的日志保留在 log 目录下: m4monitor.log... , 如果系统空间不足, 注意定时清理日志文件。

产生的 nohup.out 文件包含更明显的执行过程信息, 可以用于检查监控调度程序发生异常问题。需要定时清理该文件。

重启监控调度程序, 在 linux/unix 下 ps -ef|grep m4monitor, 找到相应的 PID 并 kill , 在执行 monitor.sh 启动; 或者通过客户端软件的“监控 -> 开始监控”重启。

4.4. 消息发送模块

“消息发送模块”将产生的告警信息通过不同方式发送给相关人员。每个监控源产生的告警消息可以定义发送给不同的用户。消息是汇总发送, 每次告警产生变化, 将当前用户相关的告警信息统一发送。

消息发送程序启动脚本(Startmsg.sh):

```
nohup    java    -cp    ./lib/ext/mssql_jdbc.jar    -Ddefault.client.encoding=GBK
-Dfile.encoding=GBK    -Duser.language=zh    -Dm4db.home=/opt/m4db    -jar
lib/m4message.jar start http://localhost:8520 > msg.out &
```

主要参数:

Start – 启动进程 或者 starttest -- 进行测试发送, 将发送所有监控对象消息

[Http://hostip:port](http://hostip:port) – 监控结果展示模块发布的地址和端口。通过 webservice 方式访问数据, 不和监控主数据库发生交互。便于部署在内外网隔离的机器, 有更好的安全性。

如果第一个参数不是 start/starttest , 则启动消息发送的配置程序:



对应配置文件 m4message.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <worktime start_time="6:30" end_time="23:0"/>
  <stage when_danger_interval_on_weekend="-1" when_danger_interval_on_workday="-1"
viobjects="" send_interval_on_workday="0">
    <specobject hostname='HOST1' objectname=' 数 据 库 状 态 ' starttime="0:0"
endtime="23:59" /></stage>
  <destinations> <dest name="sms" clazz="com.m4db.msg.smsSender"/> </destinations>
  <users>
    <user name="user1" email="user1@msn.com" subscribe="ips:10.12.10.*; ">
      <dest name="sms" id="1390000000"/> </user>
    <user name="user2" email="user2@qq.com" subscribe="ips:12.0.*;10.12.12.*;">
      <dest name="sms" id="1390000001"/> </user>
  </workdays/>
  <email port="465" server="smtp.qq.com" from="xxm4db@qq.com" password="password"
username="xxm4db" proxymode="" proxyhost="" corpname="zj xx" proxyport="0"
usessl="true"/>
</config>
```

定义的配置文件 config/m4message.xml:

- Worktime 工作时间： 定义在哪个时间范围内发送告警消息。Start_time 开始时间，end_time 结束时间。上例中告警消息在 6:30 – 23:00 之间发送。
- Stage 发送的一些策略：

when_danger_interval_on_weekend 周末（周六、周日），遇到危险的告警信息，不断发送告警消息的间隔（小时），-1 表示只发送一次；

when_danger_interval_on_workday 工作日（周一到五），遇到危险的告警信息，不断发送告警消息的间隔（小时），-1 表示只发送一次；

send_interval_on_workday 工作日（周一到五），即使没有发生告警消息的变化，发送消息用于表明发送模块是正常，间隔时间（小时），0 不发送；

viobjects 重要监控对象，目前所有对象发生异常都发送消息；

first_send 所有关注的监控对象都正常，也在每天的起始时间发生；

- **specobject** 特殊监控对象发生告警消息的时间。主要用于特别重要的监控对象，即使不在 **worktime** 范围内，也发送消息。**Hostname**，**objectname** 定义监控源，监控对象。**Starttime**，**endtime** 定义起止时间段。

- **destinations** 定义实现消息发送接口的类。接口如下：

```
public interface IMessageSender {  
    //消息发送方式名称  
    String getWayName();  
    //发送消息 message 到用户列表  
    void send(Map<String,Object> message);  
}
```

每种消息发送实现，定义为一个 **dest**，如

dest name="sms" clazz="com.m4db.msg.smsSender"

系统内置了邮件发送方式。需要配置 **email** 段内容，没有定义则不发送邮件。邮件附件内容包含更加详细的告警信息。

- **Users** 定义了消息发送给谁。

<user name="user1" email="user1@msn.com" subscribe="ips:10.12.10.*;"

定义了 **email** 地址，则发送邮件。

Subscribe 定义该用户接收哪些消息。可以使用正则表达式，每个子项以;分隔。

有两种类型 **IPS**，按 IP 地址匹配；**cat** 按监控源大类匹配：**db, os, net, web** 4 种，可以写成如：**cat:os;db**。

用户的 **dest** 段定义每个发送方式的用户 ID，如

<dest name="sms" id="1390000000"/>

- **Email** 段 定义邮件发送的一些配置。邮件服务器的端口(**port**)，邮件服务器地址(**server**)，发送者的邮箱(**from**)，口令 (**password**)，用户名(**username**)，使用代理的信息(**proxymode:socks, proxyhost, proxyport**)，是否使用 SSL(**usessl: true/false**)，邮件内容中组织名称 (**corpname**)。

4.5.代理模块

“代理模块”，主要用于监控源无法通过提供的协议（**ssh/telnet, jdbc**）连接，部署在监控源上的程序。采集监控需要的文本，执行客户端软件发送的一些指令等。

Agent 目录下执行（**m4agent.sh/m4agent.bat**）：

java -jar m4agent.jar 启动代理程序。

首先装载 **m4ageng.xml** 配置文件：

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<config>  
    <agent          port="6780"          interval="0"          logpath="/c:/m4db/agent/log"  
    scriptpath="/c:/m4db/agent/script"/>
```

- **Agent** 段 **port**，代理程序的监听端口。**Logpath** 监控调度模块获取监控对象数据的目录；**Scriptpath** 客户端软件执行命令时，默认的脚本路径；
- **Objects** 段 定义哪些监控对象。**Name** 监控对象的名称，**filename** 获取数据的文件；**lastvisit** 监控调度模块上次访问数据的时间，**lastmodified** 文件的修改时间
- **Clients** 段定义了哪些机器可以访问该监控源 可以通过客户端软件定义监控源是“测试连接”进行注册。

监控调度程序每次采集监控对象数据，代理端日志信息如下：

2018-1-9 16:20:40 com.m4db.command.GetCommand execute

信息: [object:ogg.out] last visit:2018-01-06 16:18:39, [file:/c:/m4db/agent/log\ogg.out] last modified:2018-01-06 16:18:54

5. 监控对象定义

M4DB 系统的监控对象定义方便、灵活，不需要二次开发过程。

定义可以从预定义对象复制、其他监控源的定义等地方复制；

复制可以是整个监控源复制，同时选择多个监控源等方式。

操作说明在 [4.1.8 配置 -> 监控对象](#) 进行了说明。下面分别对不同的连接方式进行定义的区别详细介绍。

5.1 SSH/TELNET 连接

通过 Ssh/telnet 方式连接到主机，执行的脚本可以是普通的命令，或者 shell 的脚本文件。主要对命令执行返回的文本进行解析处理。

5.1.1) 执行脚本的定义

- 简单的脚本，如检查 OGG 的接收进程：

```
ps -ef|grep server|grep goldengate
```

- 复杂点的脚本，如检查 OGG 状态：

```
cd /goldegnate/gg
```

```
./ggsci <<EOF
```

```
info all
```

```
exit
```

```
EOF
```

- 脚本文件定义在监控源上，调用 shell 文件，输出结果，如

```
/goldengate/gg/ggscinfo.cmd
```

该脚本文件位于监控源主机上，内容和前面的“检查 OGG 状态”一样。

5.1.2) 结果解析的定义

- 列分割符：

Ssh/telnet 产生的结果输出是文本内容。对文本的解析是逐行、分隔为列进行处理。

列分割符 文本的分割字符 默认使用的空格 tab 进行分隔。特殊的字符使用\转义，如\|，以管道符分隔。

列分隔出来，名称对应的列名/列数 ，如果结果的项目是多行记录，需要区分它们。选取能区分的列，如进程信息是 PID，文件系统是挂载点等。

- 有效文本选取：

有效行开始定义 有效行结束定义

文本的有效内容可以通过 有效行结束定义 行数 进行过滤。

行数
有效的匹配字符串
开始行的选项：

行数，用于跳过前面的几行数据；

有效的匹配字符串，只解析包含匹配字符串的行；

开始于匹配的字符串，找到匹配的字符串后，下一行开始进行解析。

行数
行数
结束行的选项：

行数，指定到哪行不再解析结果，填写 99999，表示不做限制。

结束于匹配的字符串，找到匹配的字符串后，不再解析。

5.1.3) 项目定义

■ 列名/列数:

从结果集中经过解析产生的列，可以定义为项目。如上例的 OGG 状态检查定义中，结果输出内容：

```
GGSCI (C:\ogg) 2>
```

Program	Status	Group	Lag at Chkpt	Time Since Chkpt
MANAGER	RUNNING			
EXTRACT	RUNNING	DZG2_IF	00:00:01	00:00:04
EXTRACT	RUNNING	EZG_CX2	00:00:00	00:00:09
EXTRACT	RUNNING	EZG_CX3	00:00:00	00:00:00

第二列是 ER 进程的状态，可以定义“项目名称”为“状态”，

“列名/列数”需要填写 2，“值类型”是字符串型。

项目属性

项目名称

列名/列数

LINE_CONTENT 表示单行内容

NAME__ 表示项目名称的值

值类型

■ 项目名称经过解析，实际输出的名称：

结果集汇总为一行的，使用“监控对象名.项目名称”；

结果集为多行的，使用“名称对应列的值.项目名称”。

■ 列名/列数 支持简单计算，填写“= xxx”，如

项目属性

项目名称

列名/列数

LINE_CONTENT 表示单行内容

NAME__ 表示项目名称的值

值类型

文件系统检查中，定义了空间的大小， 剩余空间。定义“使用空间”项目，它的计算公式是：“大小 - 剩余”。

“测试执行”后的评估结果：

输出
评估结果:ok
评估项目:
/software.大小:ok [20]
/software.剩余:ok [4.25]
/software.使用空间:ok [15.75]

■ 结果集模式:

如果是多行的输出结果，但评估时不考虑明显数据，只进行计数，或者输出结果是相同内容。

可以定义结果集为单行模式。 结果集模式 ☒ 单行

如监控某个进程的数量，1521 端口监听的数量，采样 vmstat 几次的输出。

在定义项目时需要使用集合函数。

结果集函数

avg

sum

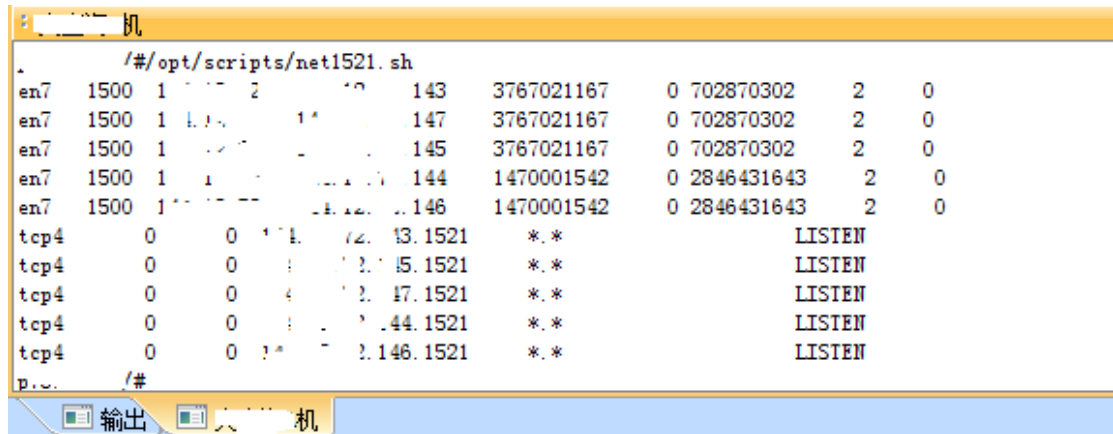
max

min

count

命令输出是多行,但对对象结果集模式是单行时

如 1521 端口监听的数量，使用 count 函数。定义了“总数”的项目，表达式为：“总数 >=10”
结果的详细输出如下：



结果解析过程，对输出行做了计数，评估的结果输出：

输出

评估结果:ok

评估项目:

RAC网络及监听. 总数:ok [10]

5.1.4) 表达式定义

表达式的作用是对解析后的结果进行评估，判断是否告警。

■ 基于行数的评估：

结果集模式

单行

多行

期望行数

==

结果集模式选“多行”，使用表达式，“期望行数” == 、 >= 、 <= 、 > 、 < 填入的值，评估实际的行数，产生告警信息。

注意：需要定义至少一个项目。

■ 项目的评估

每个监控的子项目都可以定义评估表达式。实际结果输出中，评估结果相互不影响，如：

项目	结果	
F 2. Chkpt延迟		00:01:20
R 2. 延迟		00:49:22
R 3. Chkpt延迟		00:00:45
l 3. 延迟		00:39:52
F 1. Chkpt延迟		00:00:04
F 1. 延迟		00:00:00
1. 状态		RUNNING

正常值表达式	状态 == "RUNNING"
异常值表达式	
危险值表达式	状态 == "ABENDED"

项目的评估表达式可以定义 3 种：

先评估“危险值表达式”，如果为真，该项目的告警信息是危险状态，不再进行下面的评估；

再评估“异常值表达式”，如果为真，该项目的告警信息是告警状态，不再进行下面的评估；

最后评估“正常值表达式”，如果为真，该项目的状态是正常，为假，项目的状态是告警。

■ 整体的评估

适用于结果集是单行模式的，只对整体的监控对象进行一次评估，评估的顺序、逻辑同上。

■ 表达式的函数

支持基本的算术、逻辑运算符(+*/%, && || != > < !=等)

[]标识日期类型 [yyyy-MM-dd hh24:mi:ss], ""标识字符类型, \$ 标识函数

对逐行文本解析提供 LINE_CONTENT 的内置项目,表示单行内容

✧ last_项目名称__ 可以取[项目名称]的上一次值：项目名称定义为 ABC，则可以在表达式中使用 last_ABC__，获取上次采集到的值。监控系统第一次启动，没有缓存该项目值时，使用 0 值。

内部函数：\$CONTAINS(src, search) 字符串包含

\$STARTSWITH(src, search) 字符串前缀比较

\$ENDSWITH(src, search) 字符串后缀比较

\$SYSDATE 当前时间

\$DAYEQUALS(dt1, dt2) 日期比较

\$CALCDATE(date, 年, 月, 日, 时, 分, 秒) 日期计算，带入年月日等参数可以为负数；

扩展函数：

substringM4(src, start, end) 字符串截取子串

substrM4(src, start) 字符串截取子串, start 到结尾

replaceAllM4(src, search, repl) 替换字符串

replaceFirstM4(src, search, repl) 替换字符串（第一个匹配的）

hms2NumberM4(hhmiss) 将时：分：秒的字符串转为数值，以秒为单位，异常为 0

cmpStrDateM4(strdt1, strdt2) 比较两个字符串日期的大小，返回 1, 0, -1

cmpDateM4(dt1, dt2) 比较两个日期，返回 1, 0, -1

diffStrDateM4(strdt1, strdt2) 计算 2 个字符串日期的差值(毫秒)

diffDateM4(dt1, dt2) 计算 2 个期的差值(毫秒)

函数的使用，\$函数名(参数。。。)，如使用 startswith 函数判断字符串前缀：

延迟 > "00:20:00" && ! \$STARTSWITH(延迟, "unkn")

5.2 JDBC 连接

数据库连接是 JDBC 方式实现。对执行的 sql 脚本，主要是 select 语句，进行解析评估。

支持 mysql 的 show 相关命令。

名称对应的列名/列数，列名/列数，使用 select 选取出来的字段名。

列分割、有效文本不适用；结果集模式，结果集函数适用。

值类型可以选择“日期”，保存在结果中转为 YYYY-MM-DD HH24:MI:SS 字符串。

5.3 SNMP 连接

SNMP（simple network management protocol）简单网络管理协议连接主要用在网络设备或 windows 机器监控，以及其他实现了 SNMP 管理的应用。

SNMP 使用 OID（object id）获取监控结果。M4db 系统提供了“管理工具 -> SNMP-MIB 查询”功能，便于用户在定义监控对象时进行查询、测试。

采集结果的输出可以是单行内容，或者表格内容。

■ 单行结果集例子

如采集设备的系统信息，通过 SNMP-MIB 查询工具获取的信息如下：

```
h3c1
sysDesc.0 = H3C Comware Platform Software, Software Version 5.20, Release 1808P27
H3C S5800-32F
Copyright (c) 2004-2014 Hangzhou H3C Tech. Co., Ltd. All rights reserved.
sysObjectID.0 = 1.3.6.1.4.1.25506.1.339
sysUpTime.0 = 131 days, 13:43:17.61
sysContact.0 = Hangzhou H3C Tech. Co., Ltd.
sysName.0 = ZJDS-LCHJ-IN-h3c-5800
sysLocation.0 = Hangzhou, China
sysServices.0 = 78
```

对应 OID 的描述信息：

```
输出
OID = .1.3.6.1.2.1.1
名称 : system
父节点 : mib-2
节点号 : 1
访问 :
语法 :
状态 :
描述 :
类型 : 普通记录
```

则“执行脚本”中填写 1.3.6.1.2.1.1，其中 sysUptime 的 OID 信息：

```
输出
OID = .1.3.6.1.2.1.1.3
名称 : sysUpTime
父节点 : system
节点号 : 3
访问 : read-only
语法 : TimeTicks
状态 : mandatory
描述 :
"The time (in hundredths of a second) since the network management portion of the system was lastre-initialized."
类型 : 普通记录
```

可以定义监控对象 system：

结果集模式 ☒ 单行

执行脚本

1.3.6.1.2.1.1

项目属性

项目名称

sysUptime

NAME__ 表示项目名称的值

列名/列数

3

值类型

字符串

“测试执行”，后监控输出（监控结果展示界面中的原始输出内容相同）：

```
h3c1
sysDesc.0 = H3C Comware Platform Software, Software Version 5.20, Release 1808P27
H3C S5800-32F
Copyright (c) 2004-2014 Hangzhou H3C Tech. Co., Ltd. All rights reserved.
1.3.6.1.2.1.1.2.0 = 1.3.6.1.4.1.25506.1.339
1.3.6.1.2.1.1.3.0 = 131 days, 13:56:01.05
1.3.6.1.2.1.1.4.0 = Hangzhou H3C Tech. Co., Ltd.
sysName.0 = ZJDS-LCHJ-IN-h3c-5800
1.3.6.1.2.1.1.6.0 = Hangzhou, China
1.3.6.1.2.1.1.7.0 = 78
```

解析后的评估结果， 定义的项目，名称替换为有实际意义的字符：

```
输出
评估结果:ok
评估项目:
system.sysUptime:ok [131 days, 13:56:01.05]
```

如果返回结果是同类的属性，有多个子结果。如 CPU 使用率，CPU 的每个线程都有自己的返回值。可以使用 *， 标记 列名/列数， 使用结果集函数进行汇总。

■ 多行结果集例子

OID 是表格类型的返回，Table 或者 Entry 类。如网络的接口：

```
输出
OID = .1.3.6.1.2.1.2.2
名称 : ifTable
父节点 : interfaces
节点号 : 2
访问 : not-accessible
语法 : SEQUENCE OF IfEntry
状态 : mandatory
描述 :
"A list of interface entries. The number of entries is given by the value of ifNumber."
类型 : 表
```

执行脚本使用表的 OID: 1.3.6.1.2.1.2.2

执行脚本

1.3.6.1.2.1.2.2

可以使用 ifIndex 或者 ifDesc 做记录的区分，本例使用 ifDesc。ifDesc 对应的 OID:

```
OID = .1.3.6.1.2.1.2.2.1.2
```

名称 : ifDescr，所以名称对应的列名/列数 1.2 填写与执行脚

本 OID 的子节点值：1.2，定义了两个项目：

ifPhy，接口的 MAC 地址

```
OID = .1.3.6.1.2.1.2.2.1.6
名称 : ifPhysAddress
```

，项目定义：

项目属性

项目名称 NAME__ 表示项目名称的值

列名/列数 值类型

ifStatus, 接口的状态

```
OID = .1.3.6.1.2.1.2.2.1.8
名称 : ifOperStatus
父节点 : ifEntry
节点号 : 8
访问 : read-only
语法 : INTEGER {
up(1),down(2),testing(3),unknown(4),dormant(5)
```

项目定义:

项目属性

项目名称 NAME__ 表示项目名称的值

列名/列数 值类型

正常值表达式

```
h3c1
1.3.6.1.2.1.2.2.1.2.36 = GigabitEthernet1/1/5
1.3.6.1.2.1.2.2.1.2.37 = GigabitEthernet1/1/6
1.3.6.1.2.1.2.2.1.2.38 = GigabitEthernet1/1/7
1.3.6.1.2.1.2.2.1.2.39 = GigabitEthernet1/1/8
```

测试执行的监控输出, 如:

解析评估后的结果:

```
GigabitEthernet1/0/2.ifPhy:ok [70:ba:ef:6b:42:1a]
GigabitEthernet1/0/2.ifStatus:ok [1]
GigabitEthernet1/0/3.ifPhy:ok [70:ba:ef:6b:42:1b]
GigabitEthernet1/0/3.ifStatus:warning [2]
评估结果:warning GigabitEthernet1/0/4.ifPhy:ok [70:ba:ef:6b:42:1c]
评估项目: GigabitEthernet1/0/4.ifStatus:ok [1]
```

如果使用的执行脚本是 Table 的下一层级 Entry, 如上例的 OID: 1.3.6.1.2.1.2.2.1.8
则列名/列数都截断保留 1 为子节点号: ifDesc 2, ifPhy 6, ifStatus 8。

5.4 JMX 连接

JMX 是 Java Management Extensions (JAVA 管理扩展) 的缩写, 是为应用程序、设备、系统等提供植入管理功能的框架。主要应用在 java 的应用服务器, 如 weblogic, websphere 等监控。通过查询相应的 MBean (managed bean), 获取监控结果。M4db 系统提供了 JMX-MBEAN 的查询工具, 用于测试、配置监控对象。

“执行脚本” 第一行填写获取 MBEAN 的查询语句:

[域]:限定条件,...

第二行填写对返回结果的处理, 如:

N[].M.{A, B, C}

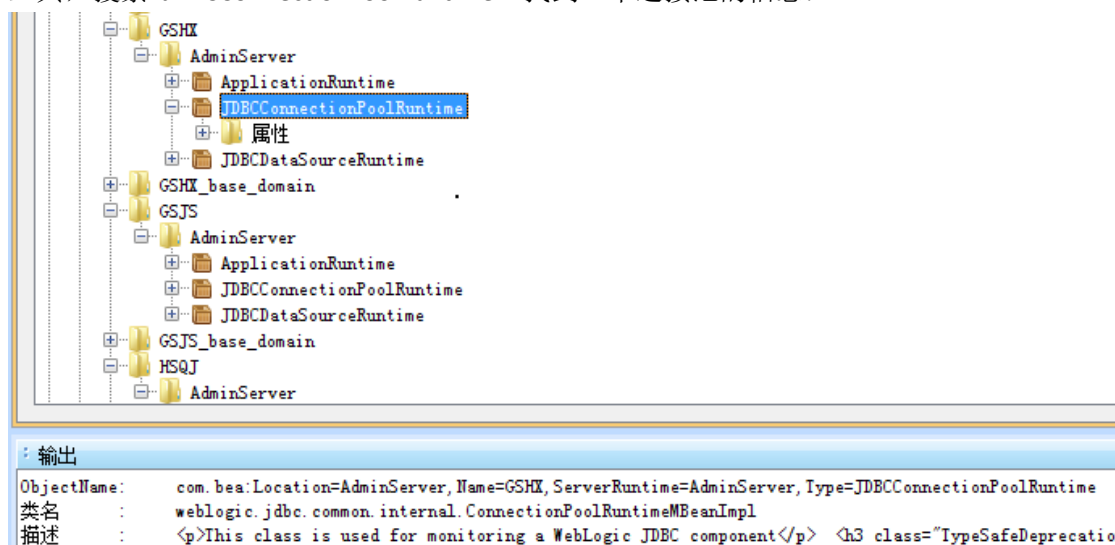
N, M 是层级, 其中[], 表示获取的是数组类型。

{ } 内是最终获取的属性。如果属性的值是复合类型, 使用<X;Y;Z;>形式获取该属性 X,Y,Z 的值。

如 A 是复合属性, 要获取其中的 X, Y 值, 填写: {A<X;Y;>}。

■ 获取连接池例子

在管理列表中选取一个以 JMX 连接的应用服务，“装载 MBean”，通过 JMX-MBEAN 查询工具，搜索：JDBCConnectionPoolRuntime。找到一个连接池的信息：



ObjectName 是可以通过 JMX 查找到该对象的查询语句:

```
com.bea:Location=AdminServer,Name=GSHX,ServerRuntime=AdminServer,Type=JDBCConne
ctionPoolRuntime
```

该系统有多个连接池，Name 限定条件可以是*，匹配所有连接池。所以定义的“执行脚本”第一行为：

```
com.bea:Location=AdminServer,Name=*,ServerRuntime=AdminServer,Type=JDBCConnectio
nPoolRuntime
```

也因此返回的结果是多个连接池信息的数组。“执行脚本”的第二行使用:[]

在 JMX-MBEAN 查询工具，查看 JDBCConnectionPoolRuntime 的属性。点击相应的属性，说明信息显示在主输出窗口中。

关注属性:

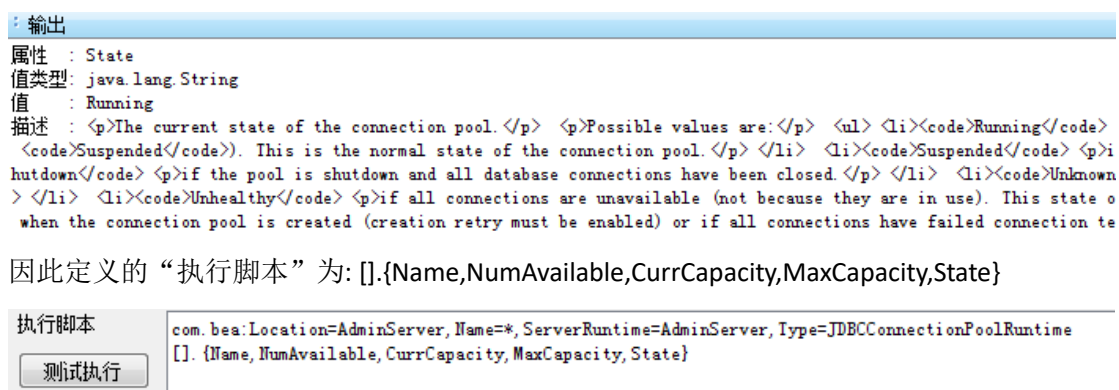
Name 连接池的名称

NumAvailable 有效的连接数

CurrCapacity 连接池的容量

MaxCapacity 连接池的最大容量

State 连接池的状态:



区别每个连接池的名称

名称对应的列名/列数

Name

项目属性

项目名称 NAME__ 表示项目名称的值

列名/列数 值类型

正常值表达式

定义项目:

项目名	列名/列数	正常状态脚本
当前容量	CurrCapacity	当前容量 < 最大容量
最大容量	MaxCapacity	
可用数	NumAvailable	
状态	State	状态 == "Running"

“测试执行”，监控输出结果:

运维服务器

```
com.bea:ServerRuntime=AdminServer,Name=HXZG,Location=AdminServer,Type=JDBCConnectionPoolRuntime={Name=HXZG, MaxCapacity=15, State=Running, NumAvailable=1, CurrCapacity=1}
```

解析评估的结果:

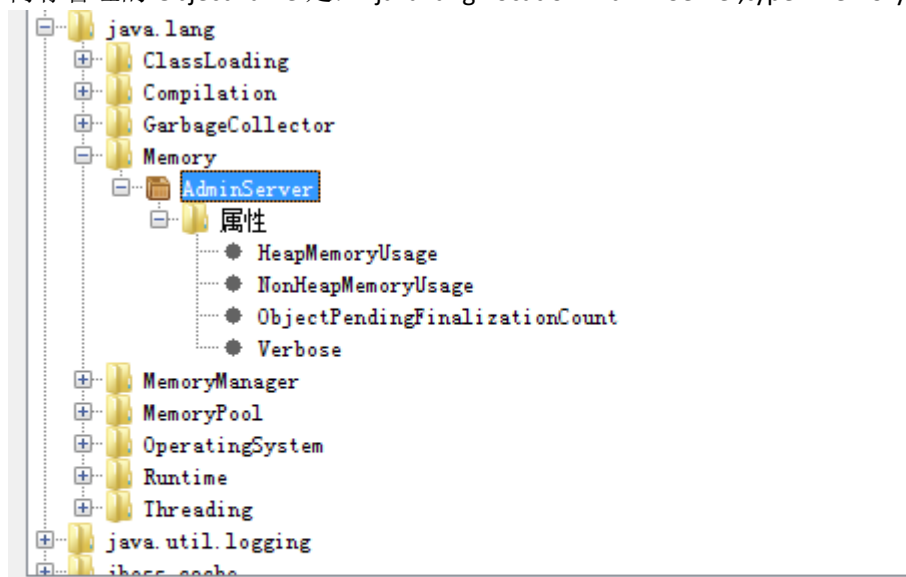
输出

```
评估结果:ok
评估项目:
WBJH.当前容量:ok [1]
WBJH.最大容量:ok [15]
WBJH.可用数:ok [1]
WBJH.状态:ok [Running]
GSHX.当前容量:ok [1]
GSHX.最大容量:ok [15]
```

■ 获取堆内存例子

在 JMX-MBEAN 查询工具中找到“Memory”属性，

内存管理的 ObjectName 是: `java.lang:Location=AdminServer,type=Memory`



输出

```
ObjectName: java.lang:Location=AdminServer,type=Memory
类名: sun.management.MemoryImpl
描述: Information on the management interface of the MBean
```


属性中的堆内存使用情况：

```
输出
属性 : HeapMemoryUsage
值类型: javax.management.openbean.CompositeData
值 : javax.management.openbean.CompositeDataSupport (compositeType=javax.management.openbean.CompositeDataSupport (itemType=javax.management.openbean.SimpleType (name=java.lang.Long)), (itemName=init, itemType=javax.management.openbean.SimpleType (name=java.lang.Long)), (itemName=max, itemType=javax.management.openbean.SimpleType (name=java.lang.Long)), (itemName=used, itemType=javax.management.openbean.SimpleType (name=java.lang.Long))), contents={committed=10602807296, init=10737418240, max=10602807296, used=1157016232})
描述 : HeapMemoryUsage
```

可以看到 HeapMemoryUsage 的值类型是复合类型 CompositeData。其中包含属性：

Committed , init , max, used 。

所以“执行脚本”定义为：

```
执行脚本
测试执行
java.lang:Location=AdminServer, type=Memory
{HeapMemoryUsage<max;used>}
```

获取堆内存的复合属性中的 max, used。

定义项目：

项目属性

项目名称 最大值 NAME 表示项目名称的值

列名/列数 max 值类型 数值

项目列表

项目名	列名/列数	正常状态脚本	异常脚本	危险状态脚本	函数
最大值	max				avg
已使用	used				avg

5.5 HTTP 连接

通过 http 连接可以访问应用服务发布的网页，检测应用服务的状态。

M4db 系统提供了两个内置的变量：

CODE ， 返回代码

CONTENT，页面的内容

■ 只填写到端口

如：<http://10.12.112.6:8001>， 可以测试该端口的连通性。

可以写 <http://#ip#:端口>， #ip#，在实际采集时替换为机器定义的 IP

只定义“执行脚本”。

对象名称 页面可访问-8001 名称对应的列名/列数 1

列分割符 文本的分割字符 结果集模式 ☒ 单行 ☐ 多行 期望行

没有定义评估表达式。返回的信息是错误代码 404 （没有找到相应的服务）：

Error 404--Not Found

From RFC 2068 *Hypertext Transfer Protocol -- HTTP/1.1*:

10.4.5 404 Not Found

外部交换 1
404
监控的输出结果：

因为可以连通该端口，评估的结果任然是正常。

输出

评估结果:ok

评估项目:

如果应用服务没有启动，则无法连接，评估结果为危险。监控输出内容：

连接问题: ERROR:Connection refused

- 可以访问的页面
如 <http://10.12.112.6:8001/login.html>
可以根据返回的代码 CODE ,判断应用状态:

项目属性

项目名称 NAME__ 表示项目名称的值

列名/列数 值类型

正常值表达式

或者匹配页面中的内容来判断状态：

页面输出内容：

```
<!DOCTYPE html>
<html lang="zh-cn" ng-app="tax">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

在整体表达式定义内容匹配：

正常结果表达式

输出

评估结果:warning

评估项目:

评估结果异常: 页面可访问-7001.CODE:ok [OK]

定义有内容的包含：

正常结果表达式

输出

评估结果:ok

评估项目:

评估结果正常: 页面可访问-7001.CODE:ok [OK]

5.6 IPMI 连接

IPMI(Intelligent Platform Management Interface)是智能平台管理接口，适用于获取硬件状态，如温度、电压、风扇状态、电源状态等。
支持通过元件名称，实体、类型匹配等方式获取监控数据。

在“配置”-“主机”界面，测试连接时，输出结果包括了所有采集到的数据。可以查看到名称（name），实体(entity)，类型(type)包含哪些可能的值。

执行脚本填写： *或 all 获取所有数据。多个脚本用;分隔。

使用 name,entity,type 限定，如 name:CPU;type:memory。获取的属性包括： name, value, state, lowerCriticalThreshold, upperCriticalThreshold 等。具体解释：

Name 名称

Entity 实体

Type 类型

Value 值

Unit 单位

Rate 比率

Status 状态 ok：正常,invalid：无效。。。

lowerCritical, lowerNonCritical, lowerNonRecoverable , upperCritical, upperNonCritical, upperNonRecoverable 各种阈值：下临界危险阈值，下临界非危险阈值，下临界不可恢复阈值，上临界危险阈值，上临界非危险阈值，上临界不可恢复阈值

有效行开始定义：有效的匹配字符串，可以根据 name 再次过滤获取到的数据。

如定义监控 CPU 温度。如果根据 entity= Processor 获取数据，发现返回数据包含温度和电压 2 种。类型分别是 Temperature, Voltage。名称类似于 CPU* Temp, Vcpu*。可以再次根据名称过来 CPU 开始的记录。

执行脚本处填写: entity:Processor，有效行开始定义，选择“有效的匹配字符串”：CPU。

有效行开始定义

有效的匹配字符串

CPU

有效行结束定义

行数

99999

执行脚本

entity:Processor

测试执行

监控项目定义：

项目列表	
项目名	列名/列数
实体	entity
状态	state
类型	type
单位	unit
值	value

获取到的结果：

评估项目：
CPU3 Temp. 实体:ok [Processor]
CPU3 Temp. 状态:ok [ok]
CPU3 Temp. 类型:ok [Temperature]
CPU3 Temp. 单位:ok [DegreesC]
CPU3 Temp. 值:ok [32]
CPU4 Temp. 实体:ok [Processor]
CPU4 Temp. 状态:ok [ok]
CPU4 Temp. 类型:ok [Temperature]
CPU4 Temp. 单位:ok [DegreesC]
CPU4 Temp. 值:ok [33]
CPU1 Temp. 实体:ok [Processor]
CPU1 Temp. 状态:ok [ok]
CPU1 Temp. 类型:ok [Temperature]
CPU1 Temp. 单位:ok [DegreesC]
CPU1 Temp. 值:ok [32]
CPU2 Temp. 实体:ok [Processor]
CPU2 Temp. 状态:ok [ok]
CPU2 Temp. 类型:ok [Temperature]
CPU2 Temp. 单位:ok [DegreesC]
CPU2 Temp. 值:ok [35]

5.7 SOCKET 连接

Socket 连接方式适用于检测端口状态。如通过 telnet , ftp, smtp, pop3 等方式连接端口，监测端口是否存活。

执行脚本填写，如: telnet 1527，监测 1527 端口状态。

连接正常无显示，不能连接则提示“连接问题”。

5.8 代理连接

代理连接主要用在无法通过 ssh/telnet 直接连接的主机上的应用监控。

如 windows 系统上部署的 GoldenGate 软件，监控队列的状态。

监控调度程序只通过 GET 方式从代理端读取结果文本，不实际执行 GoldenGate 状态检查的脚本。这步需要通过定义 windows 上的“任务计划”功能完成，定时将检查的结果输出到代理指定的文本中（参见 4.5 代理模块 中的配置文件）。

“执行脚本” 只填写要读取的文件名称。

执行脚本

ogg.out

其他的定义同 ssh/telnet 连接方式中的定义（基于文本解析）：

项目属性	
项目名称	状态
列名/列数	2
正常值表达式	状态 == "RUNNING"

LINE_CONTENT 表示单行内容
NAME__ 表示项目名称的值
值类型 字符串