Oracle监控系统总览

Python1

> 往期专题请移步www.zhaibibei.cn
> 这是一个坚持Oracle，Python原创内容的公众号，欢迎关

这节是关于我的监控系统的整体功能

具体进度可关注我的公众号

长按识别二维码

关注公众号

数据库 Python原创内容

周一至周五更新

输入help查看公众号更多功能

历史文章请访问:www.zhaibibei.cn

输入@python搜索Python相关内容，其他同理

1.开发这套系统的初衷

1.1 快速了解一套数据库

大家有没这种感觉，不论甲方还是乙方，拿到一套数据库我们很难快速的知道他的配置，数据库状态以及性能状态

虽然我们手里有很多运维的脚本，但是无法有效的统一起来

1.2 提前定位性能瓶颈

如果你对一套系统不了解，在运维过程中我们往往是迷茫的，心里没有底的

特别是性能问题

1.3 多套数据库统一管理

虽然Oracle有他的统一管理工具，但是我想大多数还是不用的吧

1.4 练手Python

正好对于Python使用也有一段时间了，而工作上也有这种需求，所以才萌发了这个想法，根据自己实际运维中的需求来开发一套系统

在开始今天的正式想说的是这套系统只是辅助我们日常的运维，对于Oracle本身的一些工具，如awr和statpack,ash等工具我们还是需要熟练掌握的

这次的分享是对上次分享的一些改进，增加了一些新的功能

首先先列出来使用到的一些环境:

开发环境

操作系统:CentOS 7.4
Python版本 :3.6
Django版本: 1.10.5
操作系统用户:oms
linux/unix模块:paramiko
Oracle模块:cx_Oracle
SQL Server模块:pymssql
数据分析:pandas
前台展示:highcharts
数据存储:MySQL,redis

2.Django介绍

2.1 Django介绍

熟悉Python的人对于Python的主流Web框架肯定有所了解，各有各的好处，Django可以说是其中最为强大和流行的一个，其官方文档非常详细，网上也有不少中文的文档，大家可现行了解

官方网站:

https://www.djangoproject.com/

2.2 ORM框架

Django采用ORM模型处理数据库关系

对象-关系映射（Object-Relational Mapping，简称ORM），简单来说就是通过面向对象的方法来映射后端数据库

它通过 类(class)的方式定义关系型数据库的表结构

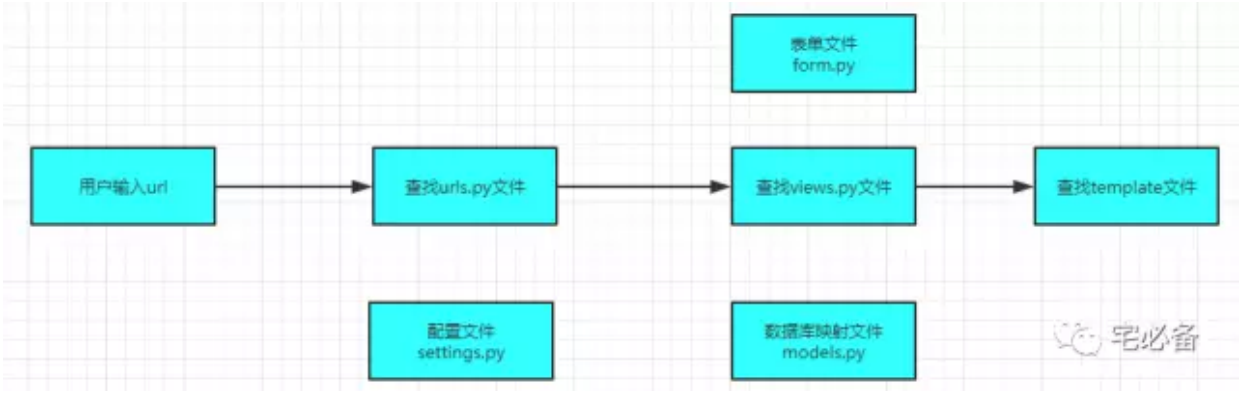
2.3 Django特性

Django 作为Web框架提供了一些非常有用的能够提升开发效率的特性

- ORM模型简化和数据库层面的沟通，如表的建立及修改
- 强大的模板(template)功能，简化前端开发难度
- form表单管理简化表单操作
- 集成了可视化管理数据库工具(admin)，免去了安装插件等动作

2.4 Django Web请求过程

接下来简单介绍一下Django如何处理用户的Web请求，以及一些常用的文件的说明



1. 首先用户输入url后,django会查找urls.py文件，找到与之对应的函数
2. urls.py对应的函数在views.py会有明确的定义，views相当于后端
3. views.py可能会调用template(模板)文件，用户在前端显示
4. model.py即前面所说的ORM模型，将数据库表定义写在该文件中
5. form.py为表单文件，Django同样提供了一套管理表单的方法
6. settings.py为配置文件，里面包含IP访问控制，插件配置以及数据库连接配置等信息

urls.py页面

```
urlpatterns = [
    url(r'^$', views.index, name='index'),
    url(r'^oracle_command/$',views.oracle_command, name='oracle_command'),
    url(r'^commandresult/$',views.commandresult, name='commandresult'),
    url(r'^oracle_status$',views.oracle_status, name='oracle_status'),
]
```

views.py

```
def oracle_status(request):
    result=oraclestatus.objects.all().order_by('tnsname')
    dic ={'result':result}
    return render_to_response('oracle_status.html',dic)
```

```
{% block content %}
<div class="col-sm-9 col-sm-offset-3 col-md-10 col-md-offset-2 main" id='content'>
<p><h3>数据库概况------每天零点更新</h3></p>
<table class="table table-hover">
<thead>
<tr>
    <th>数据库名</th>
    <th>IP地址</th>
    <th>数据库大小</th>
    <th>SGA大小</th>
    <th>表空间状态</th>
    <th>版本</th>
    <th>开启时间</th>
    <th>归档状态</th>
</tr>
</thead>
<tbody>
{% for i in result %}
            <tr>
                <td>{{i.tnsname|upper}}</td>
                <td>{{i.ipaddress|upper}}</td>
                <td>{{i.dbsize|upper}}G</td>
                <td>{{i.sga_size|upper}}M</td>
                <td>{% if i.tbstatus != "normal" %} <div style="color:#F00">{% endif %}{{i.tbstatus|upper}}</t
d>
                <td>{{i.version|upper}}</td>
                <td>{{i.startup_time|upper}}</td>
                <td>{% if i.archiver != "STARTED" %} <div style="color:#F00">{% endif %}{{i.archiver}}</td>

            </tr>
{% endfor %}
</tbody>
</div>
{% endblock %}
```

3.监控数据的获取

3.1 Linux/hp-unix

获取的内容

这里我们通过Python获取

1. Linux/HP-Unix服务器的CPU(每五分钟)
2. Linux/HP-Unix服务器的内存使用率(每五分钟)
3. Linux/HP-Unix服务器磁盘分区使用率的信息(每一小时)

获取方式

这里通过**paramiko**模块连接linux服务器

分别使用如下命令获取:

1. sar/sar
2. free/swapinfo
3. df /bdf

具体可参考如下链接:

> http://www.zhaibibei.cn/python/3.1/

3.2 Oracle数据库数据

获取的内容

这里我们通过Python获取

1. TOP SQL语句(每小时)
2. 系统状态数据 如物理读等 (每小时)
3. 等待事件(每小时)
4. 命中率信息(每小时)
5. 表空间使用情况(每天)
6. Job执行情况(每小时)

获取方式

这里通过**cx_Oracle**模块连接Oracle服务器

分别使用如下命令获取:

1. v$sqlarea
2. v$sysstat
3. v$system_event
4. v$librarycache等
5. …..

3.3 SQL Server数据获取

获取的内容

这里我们通过Python获取

1. 数据文件使用率

2. 备份情况

获取方式

这里通过**pymssql**模块连接SQL Server 服务器

分别使用如下命令获取:

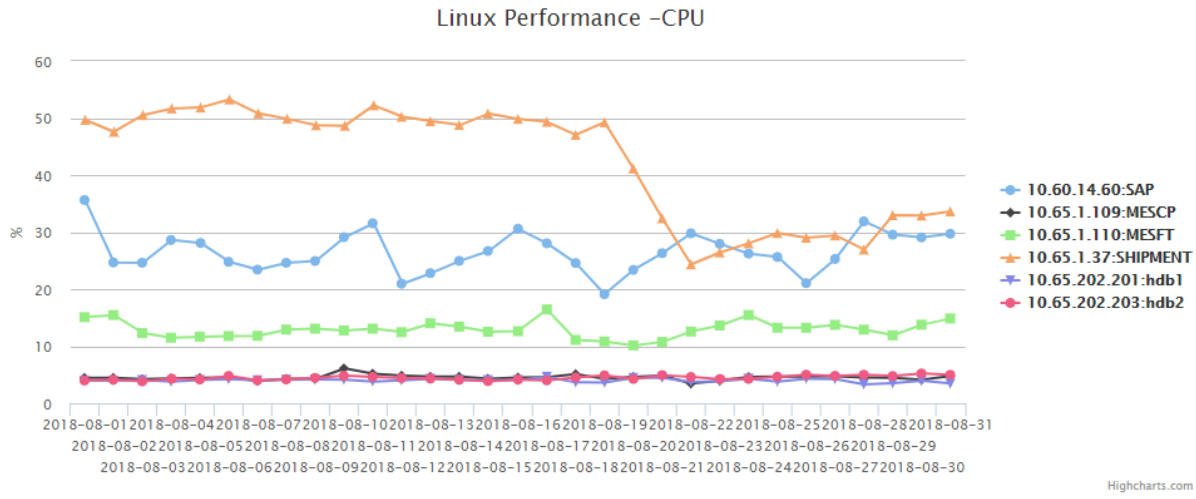
1. sp_spaceused
2. msdb.dbo.backupset

4. 监控系统的功能

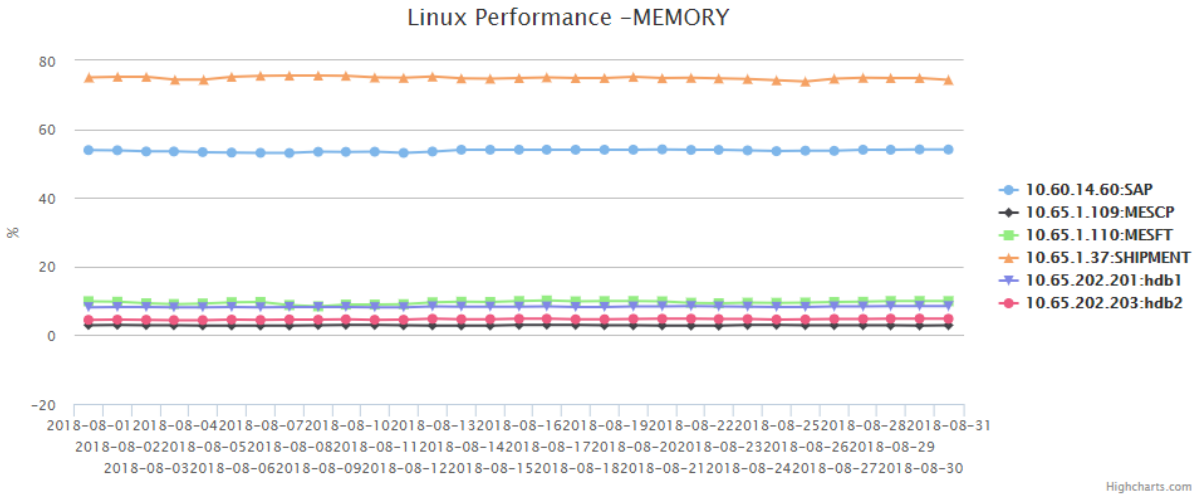
4.1 Linux/Unix监控

我们要判断一个系统是否正常，可以先从其CPU,内存来入手，这里我们获取到了服务器的数据后，可以进行分析

4.1.2 CPU趋势



4.1.3 内存趋势



4.2 Oracle监控

4.2.1 数据库巡检程序

这里我每天发送2封邮件给我，主要检查:

- 每小时redo log的产生量
- 每小时数据库的CPUTime
- 数据库每秒的硬解析次数
- 表空间的使用率
- 磁盘排序次数

```
Avg redo log switch times on edadb at 2018-08-26 are 22.46 times
CPUTime per hour on 10.65.1.113:MESASSY are 72.57 minutes since last 24 Hours
CPUTime per hour on 10.65.1.119:DCPROD are 72.73 minutes since last 24 Hours
CPUTime per hour on 10.65.1.61:ACS are 59.32 minutes since last 24 Hours
Hard parse times on 10.65.1.113:MESASSY are 70.47 times
Hard parse times on 10.65.1.61:ACS are 35.84 times
Tablespace MESFT at 10.65.1.110:MESFT is 96.08% Used
Tablespace PSAPREPSPC at 10.60.14.51:NQ1 is 91.47% Used
Tablespace PSAPSR3 at 10.60.14.70:NP1 is 99.02% Used
Tablespace SYSAUX at 10.65.202.187:SPCPROD is 92.53% Used
```

4.2.2 数据库的等待事件

这里检查每日数据库各非空闲等待事件的平均等待事件，超过一定数值则报警

```
ACS 的 control file parallel write 等待事件平均等待时间为 32.57毫秒,最近24小时共等待 38566次
ACS 的 log file switch completion 等待事件平均等待时间为 378.91毫秒,最近24小时共等待 276次
ACS 的 os thread startup 等待事件平均等待时间为 45.28毫秒,最近24小时共等待 1053次
B2BRDA 的 os thread startup 等待事件平均等待时间为 34.02毫秒,最近24小时共等待 1257次
BRAT 的 SQL*Net message from dblink 等待事件平均等待时间为 32.67毫秒,最近24小时共等待 399050次
BRAT 的 latch free 等待事件平均等待时间为 46.16毫秒,最近24小时共等待 5544次
DCPROD 的 LNS wait on SENDREQ 等待事件平均等待时间为 84.71毫秒,最近24小时共等待 24107次
DCPROD 的 PX Deq Credit: send blkd 等待事件平均等待时间为 94.53毫秒,最近24小时共等待 648389次
DCPROD 的 enq: JI - contention 等待事件平均等待时间为 185.98毫秒,最近24小时共等待 278次
DCPROD 的 enq: TX - row lock contention 等待事件平均等待时间为 63.55毫秒,最近24小时共等待 9396次
```

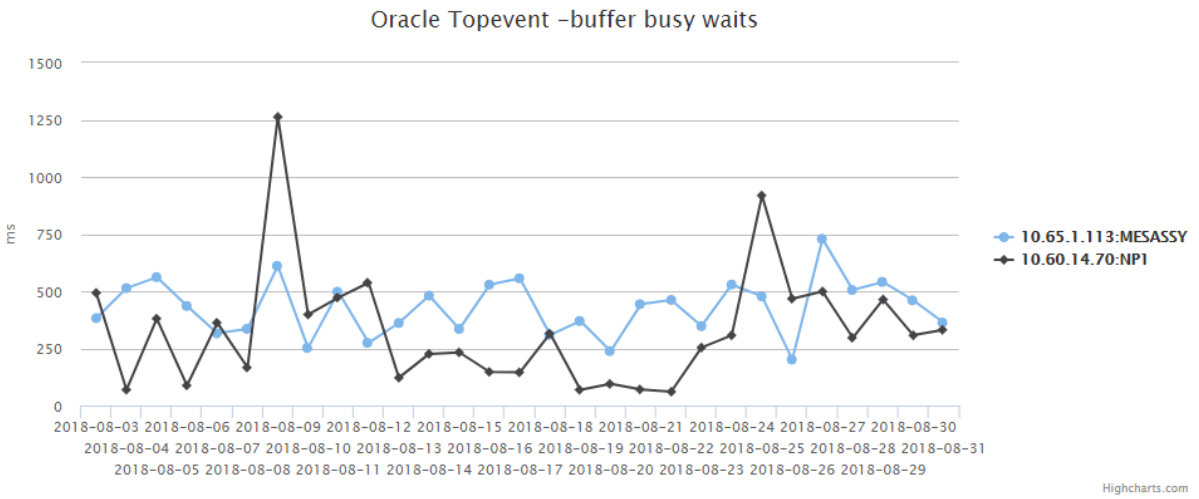
通过上面两个程序并结合CPU,内存使用率我们大体可以知道一套数据库的性能情况

接下来我们可以根据上面的问题点进行分析

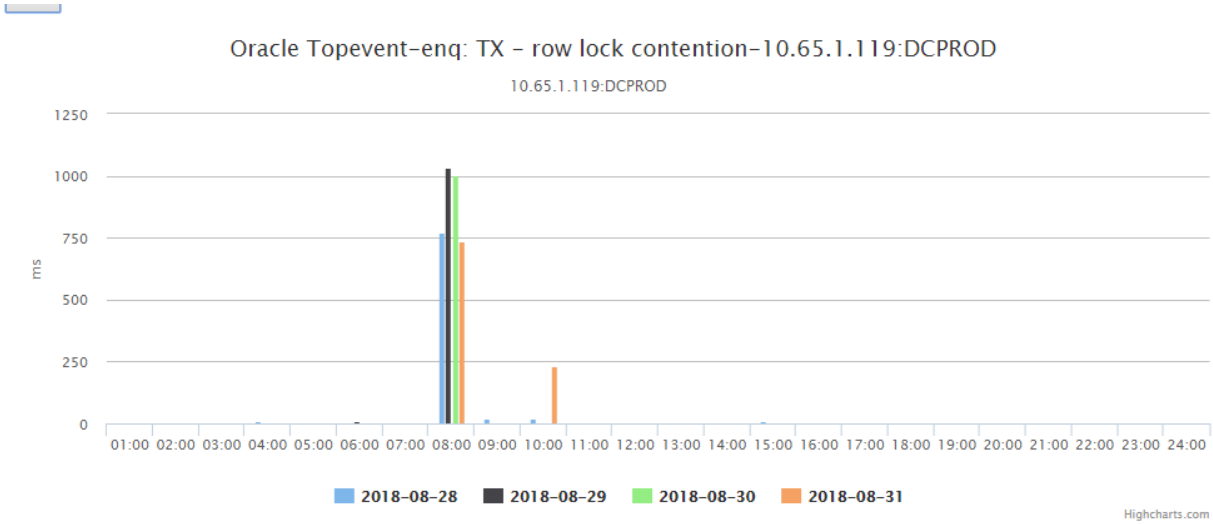
4.2.3 数据库等待事件检查

这里我们可以看到等待实践的趋势图

可以根据每天



也可以根据每小时



这样就可以对数据库的等待事件有所了解

4.2.4 数据库性能检查

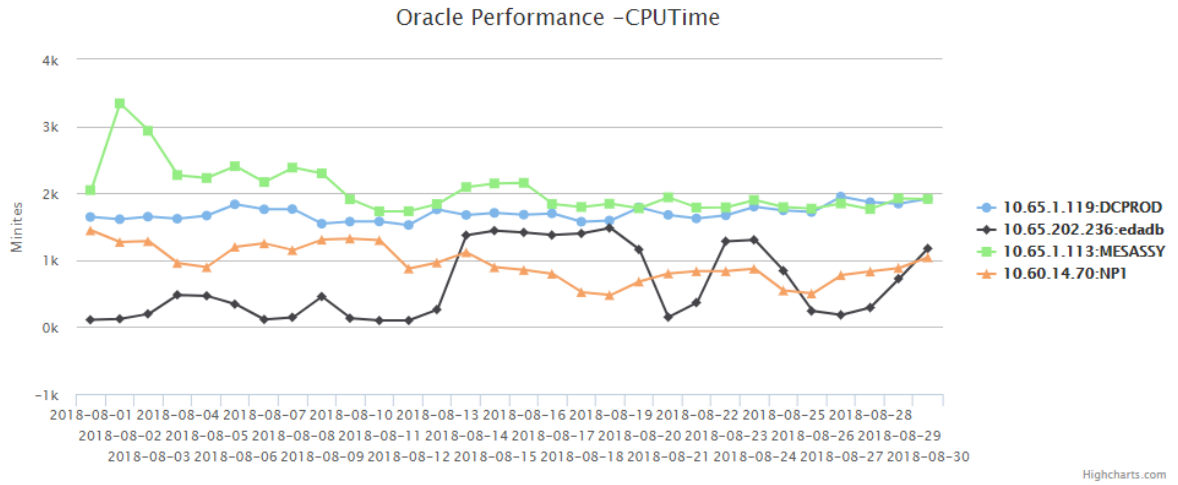
这里我们针对常见的性能指标来画出趋势图，有:

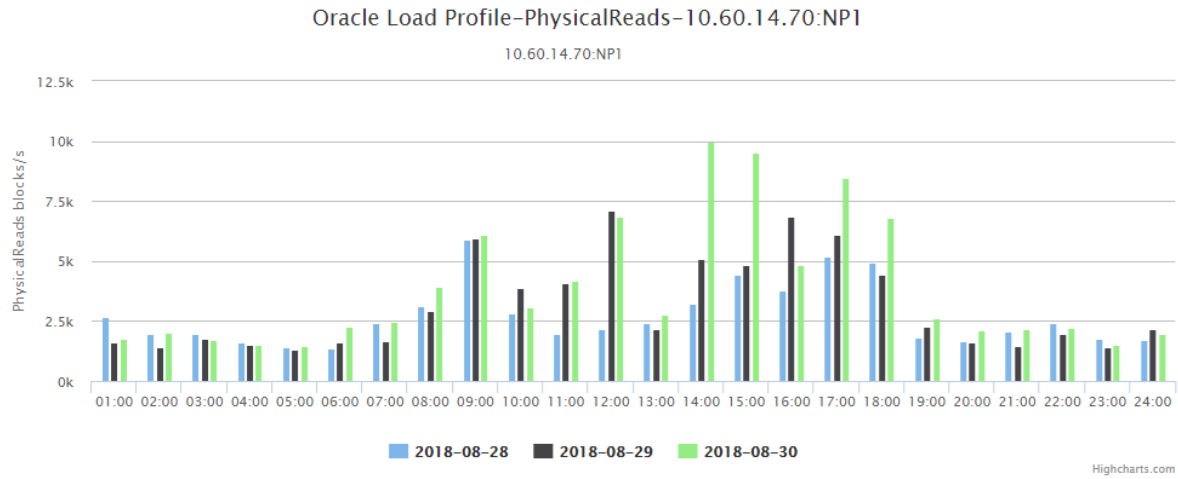
- Physical Reads
- Logical Reads
- DB Time
- CPU Time
- Hard Parse
- Total Parse
- User Commits
- User Rollbacks
- Logons

- Redo Size

等等

这里也分每天的趋势图和每小时的趋势图





所有状态一目了然

4.2.5 数据库TOPSQL检查

通过上面的检查我们可以定位到某小时某个指标较高，这时我们可以查询这个时间段的TOP SQL，主要有:

- diskreads
- buffergets
- elapsedtime
- cputime

请选择需要查询的数据库

Ipaddress: 10.65.1.113-MESASSY

Topsql type: CpuTime

Top: 10

开始时间: 20180831 09

结束时间: 20180831 10

提交

MESASSY-cputime-20180831 09-20180831 10

SQL_ID	SQL 语句	cputime	次数	平均CPU时间(S)	执行时间(S)/次	模块
2059793263	SELECT DISTINCT	59.84	2	29.92	97.2	python.exe
1603146378	update FWASSY.FWCATNS_FC_StripRecord a	54.68	66	0.83	0.9	StripRecordForFCDiebond.exe
4279886134	SELECT COUNT (*) FROM FWCATNS_STEP_RECIP	20.88	52	0.4	0.4	? @mes-ay1 (TNS V1-V3)
143213384	select * from fwcatns_product_equip wher	19.39	627	0.03	0.0	? @mes-cpft (TNS V1-V3)
4079894295	select distinct a.step as category from	13.81	661	0.02	0.0	? @mes-hdb (TNS V1-V3)
4056132497	select * from fwcatns_product_equip wher	12.09	389	0.03	0.0	? @mes-cpft (TNS V1-V3)

4.2.6 数据库命中率查询

这个功能还在评估要不要加入，暂无打算

4.2.7 数据库基线设置

可以设置数据库某个指标的某天趋势为基线方便对比

4.2.8 数据库常用命令执行

这里还有个功能就是对于我们日常用到的一些运维脚本集成到网页上方便执行

- 检查数据文件创建时间
- 检查表的分析时间
- 查看数据库段的大小
- 查看进程对应的SQL语句
- 查看会话对应的进程号
- 查看SQL的执行计划
- 检查临时表空间使用率
- 检查执行次数等于一的语句
- 检查未绑定语句

请选择数据库

ACS 10.65.1.61

请选择命令

检查数据文件创建时间

确定

数据文件创建时间-10.65.1.61-ACS

数据文件名称	文件大小	表空间	自动扩展	创建时间
/u01/app/oracle/oradata/ACS/users8.dbf	1024	USERS	NO	2018-06-22
/u01/app/oracle/oradata/ACS/users7.dbf	1024	USERS	NO	2018-05-08
/u01/acsdata/datafile/acs_idx07.dbf	2048	ACS_IDX	NO	2018-04-28
/u01/acsdata/datafile/acs_idx06.dbf	2048	ACS_IDX	NO	2018-04-11
/u01/app/oracle/oradata/ACS/users6.dbf	1024	USERS	NO	2018-01-08
/u01/acsdata/datafile/acs05.dbf	4096	ACS01	NO	2017-09-18

4.3 SQL Server监控

由于我也负责SQL Server，就也写了写它的脚本，本人了解不太深，没写太多，后续完善

主要功能有:

1. 备份监控:msdb.dbo.backupset
2. 数据文件空间管理:sp_spaceused

5. 监控程序的调用

目前用的是crontab在调用，正在学Celery + Redis，还是没有时间研究。。

6. 异常处理

我所有程序都写了异常处理模块，并重定向了标准错误输出到文件以确保所以程序异常得到及时处理

7. 监控报警机制

数据获取之后我们需要根据实际需求来进行分析然后发现问题并报警，

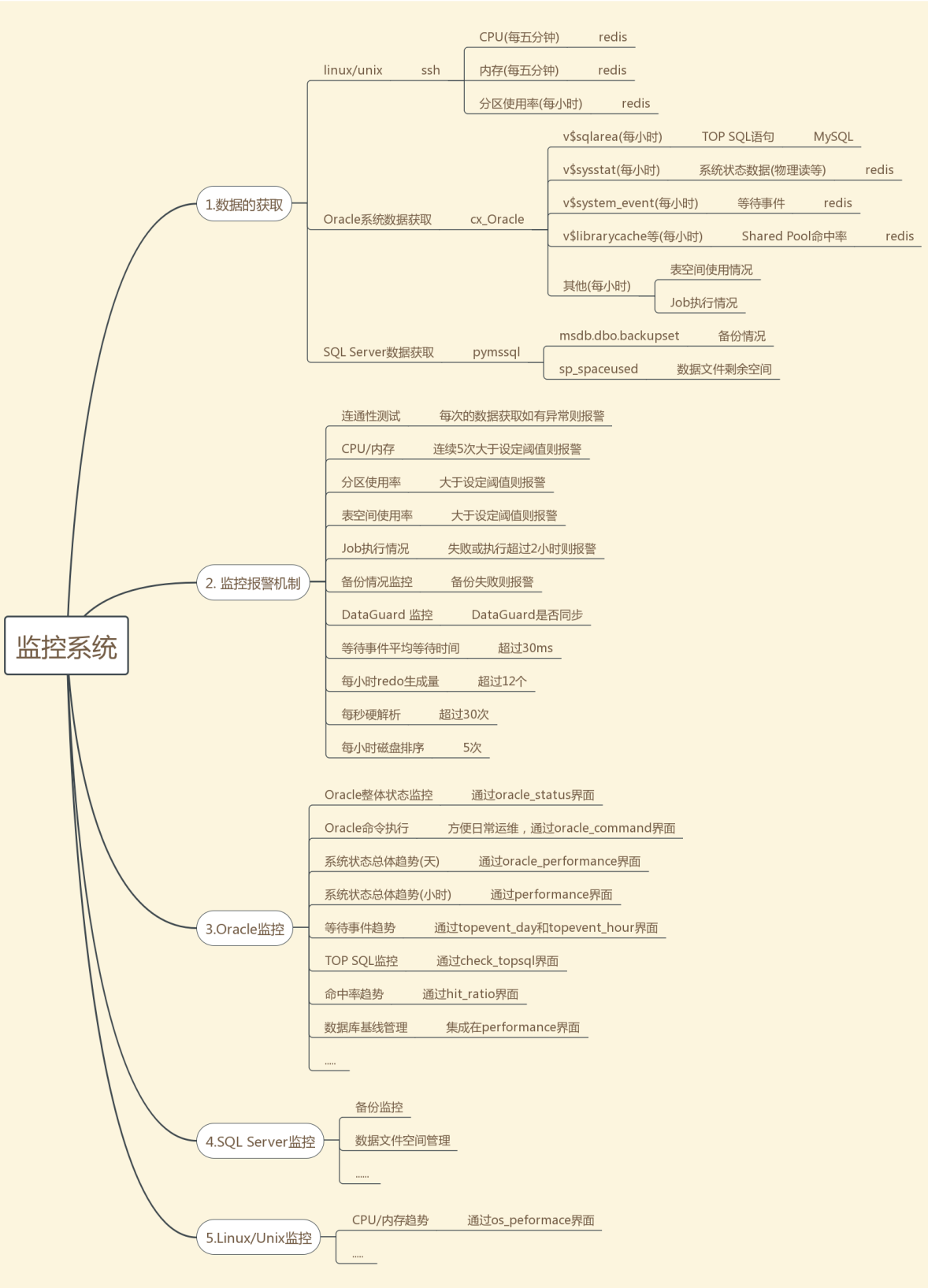
指标先紧后送，持续更改

主要有以下方面

1. 连通性测试:每次的数据获取如有异常则报警
2. CPU/内存:连续5次大于设定阈值则报警

3. 分区使用率:大于90%则报警
4. 表空间使用率:大于90%则报警
5. Job执行情况:执行失败或超过2小时报警
6. 备份情况监控:备份失败则报警
7. DataGuard 监控:DataGuard是否同步
8. Oracle alert日志有错误报警
9. 等待事件平均等待时间超过30ms报警
10. redo log 每小时超过12个报警
11. 硬解析每秒超过30次报警
12. 磁盘排序每小时超过5次报警
13. ….

好了今天的介绍就到这了，欢迎大家提问


监控系统
1.数据的获取
linux/unix ssh
CPU(每五分钟) redis
内存(每五分钟) redis
分区使用率(每小时) redis
Oracle系统数据获取 cx_Oracle
v$sqlarea(每小时) TOP SQL语句 MySQL
v$sysstat(每小时) 系统状态数据(物理读等) redis
v$system_event(每小时) 等待事件 redis
v$librarycache等(每小时) Shared Pool命中率 redis
其他(每小时)
表空间使用情况
Job执行情况
SQL Server数据获取 pymssql
msdb.dbo.backupset 备份情况
sp_spaceused 数据文件剩余空间
2. 监控报警机制
连通性测试 每次的数据获取如有异常则报警
CPU/内存 连续5次大于设定阈值则报警
分区使用率 大于设定阈值则报警
表空间使用率 大于设定阈值则报警
Job执行情况 失败或执行超过2小时则报警
备份情况监控 备份失败则报警
DataGuard 监控 DataGuard是否同步
等待事件平均等待时间 超过30ms
每小时redo生成量 超过12个
每秒硬解析 超过30次
每小时磁盘排序 5次
3.Oracle监控
Oracle整体状态监控 通过oracle_status界面
Oracle命令执行 方便日常运维，通过oracle_command界面
系统状态总体趋势(天) 通过oracle_performance界面
系统状态总体趋势(小时) 通过performance界面
等待事件趋势 通过topevent_day和topevent_hour界面
TOP SQL监控 通过check_topsql界面
命中率趋势 通过hit_ratio界面
数据库基线管理 集成在performance界面
.....
4.SQL Server监控
备份监控
数据文件空间管理
.....
5.Linux/Unix监控
CPU/内存趋势 通过os_peformace界面
.....