**概述**

**基于开源平台定制自己的达梦数据库监控产品安装及要求**

**责任数据库产品DBA：**

黄林杰 15658655447

**环境要求**

环境部署要求：

|  |  |  |  |
| --- | --- | --- | --- |
| **软件** | **版本** | **模块要求** | **备注** |
| Python | Python 3.9.0 | prometheus\_client  dmPython |  |
| Grafana | 7.4.2 |  | 页面展现平台 |
| prometheus | prometheus-2.25 |  | 监控数据存储平台 |
| dmdb-exporter.py |  |  | 达梦数据库监控采集agent |
| dmdb\_dashborad-v3.10.json |  |  | Grafana 页面展现dashboard模板 |

**监控页面**

目前监控项为：

Tps （每秒事务数）

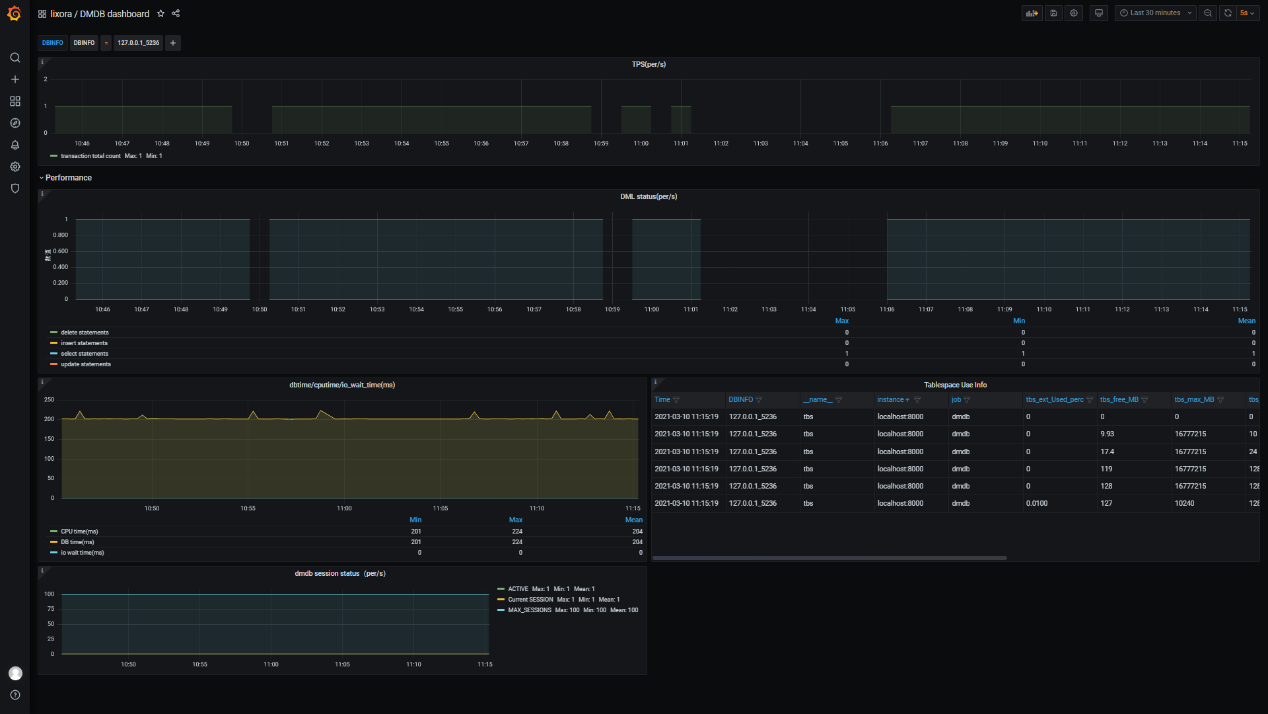
会话信息（active，inactive，maxsession，idle session）

Dml 信息（select，insert，update，delete）

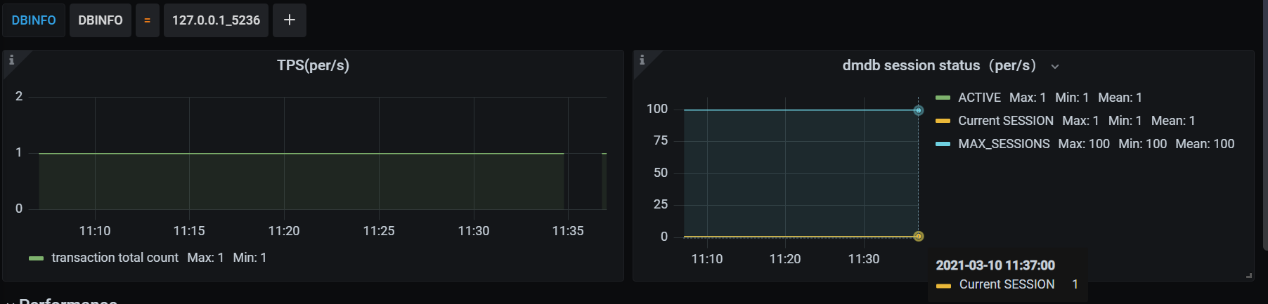
db load信息（db time cpu time,io 响应时间）

表空间信息

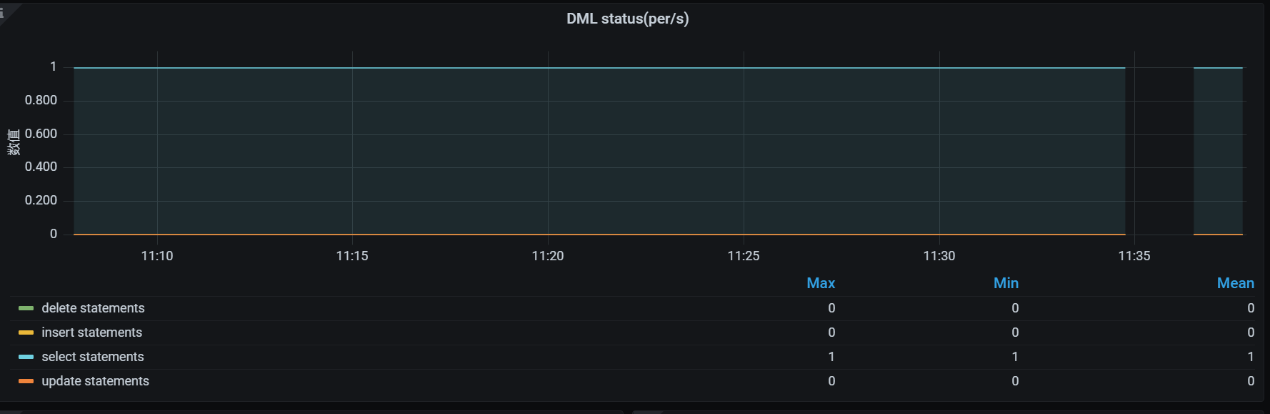
##监控页面：TPS，DML（select/insert/delete/update），dbtime，表空间监控



##监控页面：tps,会话状态



##监控页面：dml 页面





**安装及部署**

**Python 环境安装：**

1. 软件安装

2. Python 模块安装（prometheus\_client，dmPython）

**Grafana 安装及部署：**

1.软件安装;

2.Dashboard 模板导入；

略；

**Prometheus 安装及部署：**

1.软件安装;

2.prometheus.yml 配置配置；

scrape\_configs:

# The job name is added as a label `job=` to any timeseries scraped from this config.

- job\_name: 'prometheus'

# metrics\_path defaults to '/metrics'

# scheme defaults to 'http'.

static\_configs:

- targets: ['localhost:9090']

**- job\_name: 'dmdb'**

**static\_configs:**

**- targets: ['localhost:8000']**

需要配置的内容如上

略

**启动dmdb\_exporter**

**各平台默认url 访问地址**

#Prometheus

http://localhost:9090/

#grafana

http://127.0.0.1:3000/

#dmdb exporter:

http://127.0.0.1:8000/

修改dmdb-exporter.py 中要监控的达梦数据库连接信息：

**启动dmdb\_exporter**

#python dmdb-exporter.py

**回显信息如下：**

C:\Python39\python.exe D:/python-project/workspace/dmdb-monitor/dmdb-exporter.py

### Starting dmdb\_exporter v2021.3.1 --support：lixora@foxmail.com

2021-03-10 11:34:45 \*\*\*get\_session\_stat is done

2021-03-10 11:34:48 \*\*\*get\_dml\_stat is done

2021-03-10 11:34:50 \*\*\*get\_load\_stat is done

2021-03-10 11:34:52 \*\*\*get\_tps\_stat is done

2021-03-10 11:34:53 \*\*\*get\_tbs\_stat is done

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*work done\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Process finished with exit code -1

**产品代码**

###############################################################

########  dmdb-exporter.py

###############################################################

# !usr/bin/env python  
# -\*- coding:utf-8 \_\*-  
"""  
@Author:huanglinjie  
@phone：15658655447  
@File:dmdb-exporter.py  
@Time:2021/2/25 11:05  
  
#Prometheus  
http://localhost:9090/  
  
#grafana  
http://127.0.0.1:3000/  
  
  
#dmdb exporter:  
http://127.0.0.1:8000/  
  
  
"""  
import time  
from prometheus\_client import Counter, Gauge,start\_http\_server,Info  
import dmPython  
  
###lixora: debug flag  
global debug  
debug=0  
  
###dmdb\_info  
user\_name = "SYSDBA"  
passwd = "sysdba\_lixora"  
server\_name = "127.0.0.1"  
server\_port = 5236  
  
  
dmdb\_exporter\_copyrights="\n ### Starting dmdb\_exporter v2021.3.1 --support：lixora@foxmail.com \n"  
  
  
  
dbtype=str(server\_name)+'\_'+str(server\_port)  
#print(dbtype)  
  
sessionstat= Gauge('session\_stat','diffferent stat session all in one',['type','DBINFO'])  
dmlstat= Gauge('dml\_stat','diffferent sql status all in one',['type','DBINFO'])  
loadstat= Gauge('load\_stat','diffferent dbtime/cputime/iotime load all in one',['type','DBINFO'])  
tpsstat=Gauge('tps\_stat','transaction per second',['type','DBINFO'])  
  
# dbname=Gauge('dbname','db name',['dbname','DBINFO'])  
# db\_arch\_mode=Gauge('db\_arch\_mode','db archive mode',['archmode','DBINFO'])  
# db\_uptime=Gauge('db\_uptime','db up time',['uptime','DBINFO'])  
# db\_version=Gauge('db\_version','db version',['dbversion','DBINFO'])  
  
  
  
#base = Info('dmdbversion', 'Description of info')  
#base.info({'数据库名':'lixora','数据库版本': '1.2.3', '数据库端口': '8888','归档模式':'N','数据库启动时间':'aaa'})  
  
tbs = Gauge('tbs','tablespace use info',['tbsname','type','tbs\_max\_MB','tbs\_size\_MB','tbs\_used\_MB','tbs\_free\_MB','tbs\_used\_percent','tbs\_ext\_Used\_percent','DBINFO'])  
  
  
  
  
  
  
  
sql\_session='''  
(select para\_name name,para\_value value from v$dm\_ini where para\_name='MAX\_SESSIONS')  
union  
(SELECT state name ,COUNT(\*) value FROM SYS.V$SESSIONS group by state)  
union  
(SELECT 'Current SESSION',COUNT(\*) SESSIONCOUNT FROM SYS.V$SESSIONS)  
'''  
sql\_dml='''  
select name,stat\_val from v$SYSSTAT where name in ('select statements','insert statements','delete statements','update statements')  
'''  
sql\_load='''  
select name,stat\_val from v$SYSSTAT where name in ('DB time(ms)','CPU time(ms)','io wait time(ms)')  
'''  
sql\_tbs='''SELECT d.tablespace\_name "Name",  
d.contents "Type",  
to\_char(nvl(a.bytes / 1024 / 1024, 0), '99999999.9') "Total Ext Size (M)",  
to\_char(nvl(a.bytes2 / 1024 / 1024, 0), '99999999.9') "Total Size (M)",  
to\_char(nvl(a.bytes2 - nvl(f.bytes, 0), 0) / 1024 / 1024, '99999999.99') "Used (M)",  
to\_char(nvl(nvl(f.bytes, 0), 0) / 1024 / 1024, '99999999.99') "Free (M)",  
to\_char(nvl((a.bytes2 - nvl(f.bytes, 0)) / a.bytes2 \* 100, 0),'990.99') "Used %",  
to\_char(nvl((a.bytes2 - nvl(f.bytes, 0)) / a.bytes \* 100, 0),'990.99') "Ext\_Used %"  
FROM sys.dba\_tablespaces d, (SELECT tablespace\_name, SUM(greatest(BYTEs,MAXBYTES)) bytes,SUM(BYTES) bytes2 FROM dba\_data\_files GROUP BY tablespace\_name) a, (SELECT tablespace\_name, SUM(BYTES) bytes FROM dba\_free\_space GROUP BY tablespace\_name) f WHERE d.tablespace\_name = a.tablespace\_name(+) AND d.tablespace\_name = f.tablespace\_name(+) order by 8,7 '''  
sql\_tps='''  
select name,stat\_val from v$SYSSTAT where name in ('transaction total count')'''  
sql\_base='''select \* from (  
select name, arch\_mode,last\_startup\_time from v$database) ,  
(select para\_value from v$dm\_ini where para\_name='PORT\_NUM'),  
(select product\_type from v$license)'''  
# sql\_port='''  
# select para\_value from v$dm\_ini where para\_name='PORT\_NUM'''  
# sql\_version='''  
# select \* from v$version where rownum=1'''  
  
  
  
  
##直接展现结果值  
def get\_base\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_base)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dm\_cursor.fetchone()  
  
  
dm\_cursor.close()  
dm\_conn.close()  
print (aa)  
dblastuptime=aa[2]  
print(dblastuptime)  
#base = Info('dmdb\_version', 'Description of info')  
#base.info({'数据库名': aa[0], '数据库版本': aa[4], '数据库端口': aa[3], '归档模式': aa[1], 'DBINFO': dbtype})  
# dbname.labels(dbname=i[0],DBINFO=dbtype)  
# db\_arch\_mode.labels(archmode=i[1],DBINFO=dbtype)  
# db\_uptime.labels(uptime=i[2],DBINFO=dbtype)  
  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()), "\*\*\*get\_base\_stat is done")  
  
  
  
def get\_tps\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_tps)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dict(dm\_cursor.fetchall())  
  
#定时时间间隔2秒，取差值  
time.sleep(2)  
  
try:  
dm\_cursor.execute(sql\_tps)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
bb = dict(dm\_cursor.fetchall())  
  
dm\_cursor.close()  
dm\_conn.close()  
  
#print ('aa-1:',aa)  
#print ('bb-2:',bb)  
  
# 遍历字典中的每一个key  
for key in bb.keys():  
#print(bb[key]-aa[key])  
tpsstat.labels(type=key,DBINFO=dbtype).set(bb[key]-aa[key])  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()), "\*\*\*get\_tps\_stat is done")  
  
  
##直接展现结果值  
def get\_session\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_session)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dm\_cursor.fetchall()  
dm\_cursor.close()  
dm\_conn.close()  
#print (aa)  
for i in aa:  
#print (i[0])  
#print(i[1])  
sessionstat.labels(type=i[0],DBINFO=dbtype).set(i[1])  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()),"\*\*\*get\_session\_stat is done")  
  
#g.labels(hostip=host\_ip).set(cup\_use\_percent) # 本机IP传入labels，CPU使用率传入value  
  
  
  
##取2次查询差值  
def get\_dml\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_dml)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dict(dm\_cursor.fetchall())  
  
#定时时间间隔2秒，取差值  
time.sleep(2)  
  
try:  
dm\_cursor.execute(sql\_dml)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
bb = dict(dm\_cursor.fetchall())  
  
dm\_cursor.close()  
dm\_conn.close()  
  
#print ('aa-1:',aa)  
#print ('bb-2:',bb)  
  
# 遍历字典中的每一个key  
for key in bb.keys():  
#print(bb[key]-aa[key])  
dmlstat.labels(type=key,DBINFO=dbtype).set(bb[key]-aa[key])  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()), "\*\*\*get\_dml\_stat is done")  
  
  
def get\_load\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_load)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dict(dm\_cursor.fetchall())  
  
#定时时间间隔2秒，取差值  
time.sleep(2)  
  
try:  
dm\_cursor.execute(sql\_load)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
bb = dict(dm\_cursor.fetchall())  
  
dm\_cursor.close()  
dm\_conn.close()  
  
if debug:  
print ('aa-1:',aa)  
print ('bb-2:',bb)  
  
# 遍历字典中的每一个key  
for key in bb.keys():  
if debug:  
print(key)  
#print(bb[key]-aa[key])  
loadstat.labels(type=key,DBINFO=dbtype).set(bb[key]-aa[key])  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()), "\*\*\*get\_load\_stat is done")  
  
#g.labels(hostip=host\_ip).set(cup\_use\_percent) # 本机IP传入labels，CPU使用率传入value  
  
  
def get\_tbs\_stat():  
dm\_conn = dmPython.connect(user=user\_name, password=passwd, server=server\_name, port=server\_port)  
dm\_cursor = dm\_conn.cursor()  
try:  
dm\_cursor.execute(sql\_tbs)  
except Exception as e:  
print(' dm\_cursor.execute(...) Error: ', e)  
aa = dm\_cursor.fetchall()  
dm\_cursor.close()  
dm\_conn.close()  
  
#print (aa)  
for i in aa:  
# print(i[0])  
# print(i[1])  
# print(i)  
  
tbs.labels(i[0],i[1],i[2],i[3],i[4],i[5],i[6],i[7],dbtype)  
  
print(time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()),"\*\*\*get\_tbs\_stat is done")  
  
  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
start\_http\_server(8000) # 8000端口启动  
print(dmdb\_exporter\_copyrights)  
while True:  
get\_session\_stat()  
get\_dml\_stat()  
get\_load\_stat()  
get\_tps\_stat()  
  
get\_tbs\_stat()  
#get\_base\_stat()  
print('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*work done\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
  
#自定义性能指标采集循环周期，默认5秒  
time.sleep(5)

#########################################################

######       dmdb\_dashborad-v3.10.json

#########################################################

{  
"annotations": {  
"list": [  
{  
"builtIn": 1,  
"datasource": "-- Grafana --",  
"enable": true,  
"hide": true,  
"iconColor": "rgba(0, 211, 255, 1)",  
"name": "Annotations & Alerts",  
"type": "dashboard"  
}  
]  
},  
"description": "DMDB advanced monitor powered by huanglj",  
"editable": true,  
"gnetId": null,  
"graphTooltip": 0,  
"id": 4,  
"iteration": 1615341881106,  
"links": [],  
"panels": [  
{  
"datasource": "Prometheus",  
"description": "dmdb session status",  
"fieldConfig": {  
"defaults": {  
"color": {  
"mode": "palette-classic"  
},  
"custom": {  
"axisLabel": "",  
"axisPlacement": "auto",  
"barAlignment": 0,  
"drawStyle": "line",  
"fillOpacity": 10,  
"gradientMode": "none",  
"hideFrom": {  
"graph": false,  
"legend": false,  
"tooltip": false  
},  
"lineInterpolation": "linear",  
"lineWidth": 1,  
"pointSize": 5,  
"scaleDistribution": {  
"type": "linear"  
},  
"showPoints": "never",  
"spanNulls": true  
},  
"mappings": [],  
"thresholds": {  
"mode": "absolute",  
"steps": [  
{  
"color": "green",  
"value": null  
},  
{  
"color": "red",  
"value": 80  
}  
]  
},  
"unit": "short"  
},  
"overrides": []  
},  
"gridPos": {  
"h": 6,  
"w": 12,  
"x": 0,  
"y": 0  
},  
"id": 2,  
"options": {  
"graph": {},  
"legend": {  
"calcs": [  
"max",  
"min",  
"mean"  
],  
"displayMode": "list",  
"placement": "right"  
},  
"tooltipOptions": {  
"mode": "single"  
}  
},  
"pluginVersion": "7.4.2",  
"targets": [  
{  
"exemplar": false,  
"expr": "session\_stat",  
"format": "time\_series",  
"interval": "",  
"legendFormat": "{{type}}",  
"refId": "A"  
}  
],  
"timeFrom": null,  
"timeShift": null,  
"title": "dmdb session status（per/s）",  
"type": "timeseries"  
},  
{  
"datasource": "Prometheus",  
"description": "tps(per/s)",  
"fieldConfig": {  
"defaults": {  
"color": {  
"mode": "palette-classic"  
},  
"custom": {  
"axisLabel": "",  
"axisPlacement": "auto",  
"barAlignment": 0,  
"drawStyle": "line",  
"fillOpacity": 10,  
"gradientMode": "none",  
"hideFrom": {  
"graph": false,  
"legend": false,  
"tooltip": false  
},  
"lineInterpolation": "smooth",  
"lineWidth": 1,  
"pointSize": 5,  
"scaleDistribution": {  
"type": "linear"  
},  
"showPoints": "never",  
"spanNulls": false  
},  
"mappings": [],  
"thresholds": {  
"mode": "absolute",  
"steps": [  
{  
"color": "green",  
"value": null  
},  
{  
"color": "red",  
"value": 80  
}  
]  
},  
"unit": "short"  
},  
"overrides": []  
},  
"gridPos": {  
"h": 6,  
"w": 12,  
"x": 12,  
"y": 0  
},  
"id": 8,  
"options": {  
"graph": {},  
"legend": {  
"calcs": [  
"max",  
"min"  
],  
"displayMode": "list",  
"placement": "bottom"  
},  
"tooltipOptions": {  
"mode": "single"  
}  
},  
"pluginVersion": "7.4.2",  
"targets": [  
{  
"expr": "tps\_stat",  
"interval": "",  
"legendFormat": "{{type}}",  
"refId": "A"  
}  
],  
"timeFrom": null,  
"timeShift": null,  
"title": "TPS(per/s)",  
"type": "timeseries"  
},  
{  
"collapsed": false,  
"datasource": null,  
"gridPos": {  
"h": 1,  
"w": 24,  
"x": 0,  
"y": 6  
},  
"id": 16,  
"panels": [],  
"title": "Performance ",  
"type": "row"  
},  
{  
"datasource": "Prometheus",  
"description": "DML status(per/s)",  
"fieldConfig": {  
"defaults": {  
"color": {  
"mode": "palette-classic"  
},  
"custom": {  
"axisLabel": "数值",  
"axisPlacement": "auto",  
"barAlignment": 0,  
"drawStyle": "line",  
"fillOpacity": 10,  
"gradientMode": "none",  
"hideFrom": {  
"graph": false,  
"legend": false,  
"tooltip": false  
},  
"lineInterpolation": "smooth",  
"lineWidth": 1,  
"pointSize": 5,  
"scaleDistribution": {  
"type": "linear"  
},  
"showPoints": "never",  
"spanNulls": false  
},  
"mappings": [],  
"thresholds": {  
"mode": "absolute",  
"steps": [  
{  
"color": "green",  
"value": null  
},  
{  
"color": "red",  
"value": 80  
}  
]  
},  
"unit": "short"  
},  
"overrides": []  
},  
"gridPos": {  
"h": 10,  
"w": 24,  
"x": 0,  
"y": 7  
},  
"id": 4,  
"options": {  
"graph": {},  
"legend": {  
"calcs": [  
"max",  
"min",  
"mean"  
],  
"displayMode": "table",  
"placement": "bottom"  
},  
"tooltipOptions": {  
"mode": "single"  
}  
},  
"pluginVersion": "7.4.2",  
"targets": [  
{  
"exemplar": false,  
"expr": "dml\_stat",  
"instant": false,  
"interval": "",  
"legendFormat": "{{type}}",  
"refId": "A"  
}  
],  
"timeFrom": null,  
"timeShift": null,  
"title": "DML status(per/s)",  
"type": "timeseries"  
},  
{  
"datasource": "Prometheus",  
"description": "dbtime/cputime/io\_wait\_time(ms)",  
"fieldConfig": {  
"defaults": {  
"color": {  
"mode": "palette-classic"  
},  
"custom": {  
"axisLabel": "",  
"axisPlacement": "auto",  
"barAlignment": 0,  
"drawStyle": "line",  
"fillOpacity": 10,  
"gradientMode": "none",  
"hideFrom": {  
"graph": false,  
"legend": false,  
"tooltip": false  
},  
"lineInterpolation": "linear",  
"lineWidth": 1,  
"pointSize": 5,  
"scaleDistribution": {  
"type": "linear"  
},  
"showPoints": "never",  
"spanNulls": true  
},  
"mappings": [],  
"thresholds": {  
"mode": "absolute",  
"steps": [  
{  
"color": "green",  
"value": null  
},  
{  
"color": "red",  
"value": 80  
}  
]  
},  
"unit": "short"  
},  
"overrides": []  
},  
"gridPos": {  
"h": 10,  
"w": 12,  
"x": 0,  
"y": 17  
},  
"id": 6,  
"options": {  
"graph": {},  
"legend": {  
"calcs": [  
"min",  
"max",  
"mean"  
],  
"displayMode": "table",  
"placement": "bottom"  
},  
"tooltipOptions": {  
"mode": "single"  
}  
},  
"pluginVersion": "7.4.2",  
"targets": [  
{  
"expr": "load\_stat",  
"interval": "",  
"legendFormat": "{{type}}",  
"refId": "A"  
}  
],  
"timeFrom": null,  
"timeShift": null,  
"title": "dbtime/cputime/io\_wait\_time(ms)",  
"type": "timeseries"  
},  
{  
"cacheTimeout": null,  
"datasource": "Prometheus",  
"description": "Tablespace Use Info",  
"fieldConfig": {  
"defaults": {  
"color": {  
"mode": "thresholds"  
},  
"custom": {  
"align": null,  
"displayMode": "auto",  
"filterable": true  
},  
"mappings": [],  
"thresholds": {  
"mode": "absolute",  
"steps": [  
{  
"color": "green",  
"value": null  
},  
{  
"color": "#EAB839",  
"value": 100  
}  
]  
}  
},  
"overrides": []  
},  
"gridPos": {  
"h": 10,  
"w": 12,  
"x": 12,  
"y": 17  
},  
"id": 14,  
"interval": null,  
"links": [],  
"options": {  
"frameIndex": 1,  
"showHeader": true,  
"sortBy": [  
{  
"desc": false,  
"displayName": "instance"  
}  
]  
},  
"pluginVersion": "7.4.2",  
"targets": [  
{  
"exemplar": false,  
"expr": "tbs",  
"format": "table",  
"instant": true,  
"interval": "",  
"intervalFactor": 1,  
"legendFormat": "",  
"refId": "A"  
}  
],  
"title": "Tablespace Use Info",  
"type": "table"  
}  
],  
"refresh": "5s",  
"schemaVersion": 27,  
"style": "dark",  
"tags": [],  
"templating": {  
"list": [  
{  
"datasource": "",  
"description": "display different dmdb server status",  
"error": null,  
"filters": [  
{  
"condition": "",  
"key": "DBINFO",  
"operator": "=",  
"value": "127.0.0.1\_5236"  
}  
],  
"hide": 0,  
"label": "DBINFO",  
"name": "DBINFO",  
"skipUrlSync": false,  
"type": "adhoc"  
}  
]  
},  
"time": {  
"from": "now-30m",  
"to": "now"  
},  
"timepicker": {},  
"timezone": "",  
"title": "DMDB dashboard",  
"uid": "ZYKex2yMk",  
"version": 44  
}