# 数据质量监控

## 概述

随着大数据时代的带来，数据的应用也日趋繁茂，越来越多的应用和服务都基于数据而建立，数据的重要性不言而喻。而且，数据质量是数据分析和数据挖掘结论有效性和准确性的基础，也是这一切的数据驱动决策的前提！如何保障数据质量，确保数据可用性是每一位数据人都不可忽略的重要环节。

## 数仓项目的数据质量监控

我们的数仓项目主要是监视以下几个数据指标：

1. 原始数据层采集的数据，每日环比和同比增长不能超过一定范围
2. 维度层id空值，重复值不能出现；
3. 数据明细层不能出现空值id；
4. 用户浏览页面的时间和订单金额的异常值（太大或者太小）不能过多。
5. 基于以上需求，首先编写通用指标查询脚本：

[atguigu@hadoop102 ~]$ vim /home/atguigu/bin/table\_check.sh

#!/usr/bin/env bash

# -\*- coding: utf-8 -\*-

# 脚本参数说明如下：

# -t 表名

# -d 日期

# -c 列名

# -n 指标名

# -v 指标对应的参数，用冒号分隔

# 指标 指标对应参数

# null\_id min:max

# dup min:max

# range range\_min:range\_max:min:max

# day\_on\_day min:max

# week\_on\_week min:max

# std\_dev min:max

# 当对应指标超出min：max范围，会触发告警

while getopts "t:d:c:n:v:" arg; do

case $arg in

t)

TABLE=$OPTARG

;;

d)

DT=$OPTARG

;;

c)

COL=$OPTARG

;;

n)

NORM=$OPTARG

;;

v)

VALUE=$OPTARG

;;

?)

echo "unkonw argument"

exit 1

;;

esac

done

HIVE\_DB=gmall

HIVE\_ENGINE=hive

mysql\_user="root"

mysql\_passwd="000000"

mysql\_host="hadoop102"

mysql\_DB="test"

mysql\_tbl="ind"

#检查空值

function null\_id() {

array=(${VALUE//:/ })

MIN=${array[0]}

MAX=${array[1]}

RESULT=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt='$DT' and $COL is null;")

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'null\_id', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

#值域检查

function range() {

array=(${VALUE//:/ })

RANGE\_MIN=${array[0]}

RANGE\_MAX=${array[1]}

MIN=${array[2]}

MAX=${array[3]}

RESULT=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt='$DT' and $COL not between $RANGE\_MIN and $RANGE\_MAX;")

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'range', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

#重复值检查

function dup() {

array=(${VALUE//:/ })

MIN=${array[0]}

MAX=${array[1]}

RESULT=$($HIVE\_ENGINE -e "select count(1) from (select $COL from $HIVE\_DB.$TABLE where dt='$DT' group by $COL having count($COL)>1) t1;")

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'duplicate', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

#数据量同比增长检查

function day\_on\_day() {

array=(${VALUE//:/ })

MIN=${array[0]}

MAX=${array[1]}

YESTODAY=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt=date\_add('$DT',-1);")

TODAY=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt='$DT';")

if [ $YESTODAY -ne 0 ]; then

RESULT=$(awk "BEGIN{print ($TODAY-$YESTODAY)/$YESTODAY\*100}")

else

RESULT=10000

fi

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'day\_on\_day', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

#数据量环比增长检查

function week\_on\_week() {

array=(${VALUE//:/ })

MIN=${array[0]}

MAX=${array[1]}

LASTWEEK=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt between date\_add('$DT',-13) and date\_add('$DT',-7);")

THISWEEK=$($HIVE\_ENGINE -e "select count(1) from $HIVE\_DB.$TABLE where dt between date\_add('$DT',-6) and '$DT';")

if [ $LASTWEEK -ne 0 ]; then

RESULT=$(awk "BEGIN{print ($THISWEEK-$LASTWEEK)/$LASTWEEK\*100}")

else

RESULT=10000

fi

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'week\_on\_week', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

#数据标准差检查

function std\_dev() {

array=(${VALUE//:/ })

MIN=${array[0]}

MAX=${array[1]}

RESULT=$($HIVE\_ENGINE -e "select std($COL) from $HIVE\_DB.$TABLE where dt='$DT';")

mysql -h"$mysql\_host" -u"$mysql\_user" -p"$mysql\_passwd" \

-e"INSERT INTO $mysql\_DB.$mysql\_tbl VALUES('$DT', '$TABLE', 'std\_dev', $RESULT, $MIN, $MAX, '$COL')

ON DUPLICATE KEY UPDATE norm\_value=$RESULT, norm\_value\_min=$MIN, norm\_value\_max=$MAX, comm='$COL';"

}

case $NORM in

"null\_id"|"range"|"dup"|"day\_on\_day"|"week\_on\_week"|"std\_dev")

$NORM

;;

esac

1. 初始化数据库：

[atguigu@hadoop103 ~]$ mysql -uroot -p000000 -e"drop database if exists test;

create database test;

CREATE TABLE test.`ind` (

`dt` date NOT NULL COMMENT '日期',

`tbl` varchar(20) NOT NULL COMMENT '表名',

`norm` varchar(20) NOT NULL COMMENT '指标名称',

`norm\_value` double DEFAULT NULL COMMENT '值',

`norm\_value\_min` double DEFAULT NULL COMMENT '下限',

`norm\_value\_max` double DEFAULT NULL COMMENT '上限',

`comm` varchar(20) DEFAULT NULL COMMENT '备注',

PRIMARY KEY (`dt`,`tbl`,`norm`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;"

1. 编写数据监控脚本：

[atguigu@hadoop102 ~]$ vim /home/atguigu/bin/check\_all\_tables.sh

#!/usr/bin/env bash

CHECK\_SHELL="table\_check.sh"

DT=$1

if [ -z $DT ]

then DT=$(date -d '-1 day' +%F)

fi

# ODS

$CHECK\_SHELL -t ods\_order\_refund\_info -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_order\_detail -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_order\_info -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_user\_info -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_payment\_info -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_refund\_payment -d "$DT" -n day\_on\_day -c id -v -10:10

$CHECK\_SHELL -t ods\_order\_refund\_info -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_order\_detail -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_order\_info -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_user\_info -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_payment\_info -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_refund\_payment -d "$DT" -n week\_on\_week -c id -v -50:50

$CHECK\_SHELL -t ods\_order\_info -d "$DT" -n range -c final\_amount -v 0:100000:0:100

# DIM

$CHECK\_SHELL -t dim\_activity\_rule\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dim\_coupon\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dim\_sku\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dim\_user\_info -d "9999-99-99" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dim\_activity\_rule\_info -d "$DT" -n dup -c id -v 0:5

$CHECK\_SHELL -t dim\_coupon\_info -d "$DT" -n dup -c id -v 0:5

$CHECK\_SHELL -t dim\_sku\_info -d "$DT" -n dup -c id -v 0:5

$CHECK\_SHELL -t dim\_user\_info -d "9999-99-99" -n dup -c id -v 0:5

# DWD

$CHECK\_SHELL -t dwd\_order\_detail -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dwd\_order\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dwd\_order\_refund\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dwd\_payment\_info -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dwd\_refund\_payment -d "$DT" -n null\_id -c id -v 0:10

$CHECK\_SHELL -t dwd\_order\_info -d "$DT" -n dup -c id -v 0:5

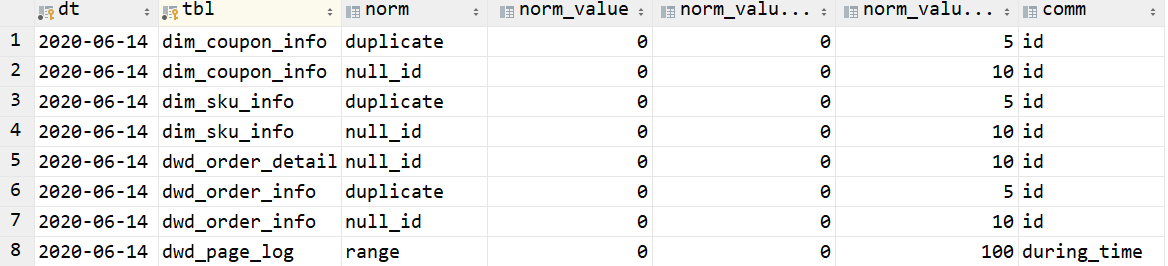
$CHECK\_SHELL -t dwd\_page\_log -d "$DT" -n range -c during\_time -v 1000:30000:0:100

添加执行权限，并执行：

[atguigu@hadoop102 ~]$ chmod +x /home/atguigu/bin/check\_all\_tables.sh

[atguigu@hadoop102 ~]$ check\_all\_tables.sh 2020-06-14

1. 等待脚本执行完毕后，观察mysql数据库，是否能看到数据：



1. 安装mysql-connector-python

sudo rpm -Uvh https://downloads.mysql.com/archives/get/p/29/file/mysql-connector-python-2.1.7-1.el7.x86\_64.rpm

1. 编写告警脚本，查询异常数据并发送邮件通知：

[atguigu@hadoop102 ~]$ vim /home/atguigu/bin/check\_notification.py

# -\*- coding: utf-8 -\*-

import mysql.connector

import sys

import smtplib

from email.mime.text import MIMEText

from email.header import Header

import datetime

def get\_yesterday():

today = datetime.date.today()

oneday = datetime.timedelta(days=1)

yesterday = today - oneday

return yesterday

def mail\_alert(message):

mail\_host = "smtp.126.com"

mail\_user = "user@126.com"

mail\_pass = "password"

sender = mail\_user

receivers = [mail\_user]

mail\_content = MIMEText(''.join(['<html>', '<br>'.join(message), '</html>']), 'html', 'utf-8')

mail\_content['from'] = sender

mail\_content['to'] = receivers[0]

mail\_content['Subject'] = Header('数据监控错误', 'utf-8')

try:

smtp = smtplib.SMTP\_SSL()

smtp.connect(mail\_host, 465)

smtp.login(mail\_user, mail\_pass)

content\_as\_string = mail\_content.as\_string()

smtp.sendmail(sender, receivers, content\_as\_string)

except smtplib.SMTPException as e:

print e

def main(argv):

# 如果没有传入日期参数，将日期定为昨天

if len(argv) >= 2:

dt = argv[1]

else:

dt = str(get\_yesterday())

# 初始化警告正文

alert\_string = []

# 查询所有指标错误的记录

connect = mysql.connector.connect(user="root", password="000000", host="hadoop102", database="test")

cursor = connect.cursor()

query = ("select tbl, norm, comm, norm\_value, norm\_value\_min, norm\_value\_max from ind "

"where dt = '" + dt + "' and norm\_value not between norm\_value\_min and norm\_value\_max")

cursor.execute(query)

for (tbl, norm, comm, norm\_value, norm\_value\_min, norm\_value\_max) in cursor:

alert\_string.append('%s表异常，指标%s值为%s，超出范围%s-%s，参考信息%s。' % (

str(tbl), str(norm), norm\_value, norm\_value\_min, norm\_value\_max, str(comm)))

# 如果警告数量大于0，发送警告

if len(alert\_string) > 0:

mail\_alert(alert\_string)

if \_\_name\_\_ == "\_\_main\_\_":

main(sys.argv)

添加执行权限，并执行：

[atguigu@hadoop102 ~]$ chmod +x /home/atguigu/bin/check\_notification.py

[atguigu@hadoop102 ~]$ check\_notification.py 2020-06-14

观察邮箱是否能收到错误报警邮件。