网站访问日志分析

1. 日志数据

网站访问日志是用户在访问网站服务器时产生的日志，它包含了各种原始信息，一般以.log结尾。通过它就可以清楚的知道用户的IP，访问时间，请求链接，请求状态，请求字节数，来源链接，用户操作系统，浏览器内核，浏览器名称，浏览器版本等等信息。对网站日志的分析统计可以使我们了解网站当前的一些状况，为网站的各种优化升级甚至公司营销策略提供依据。

本实验的数据包含了两个文本文件：

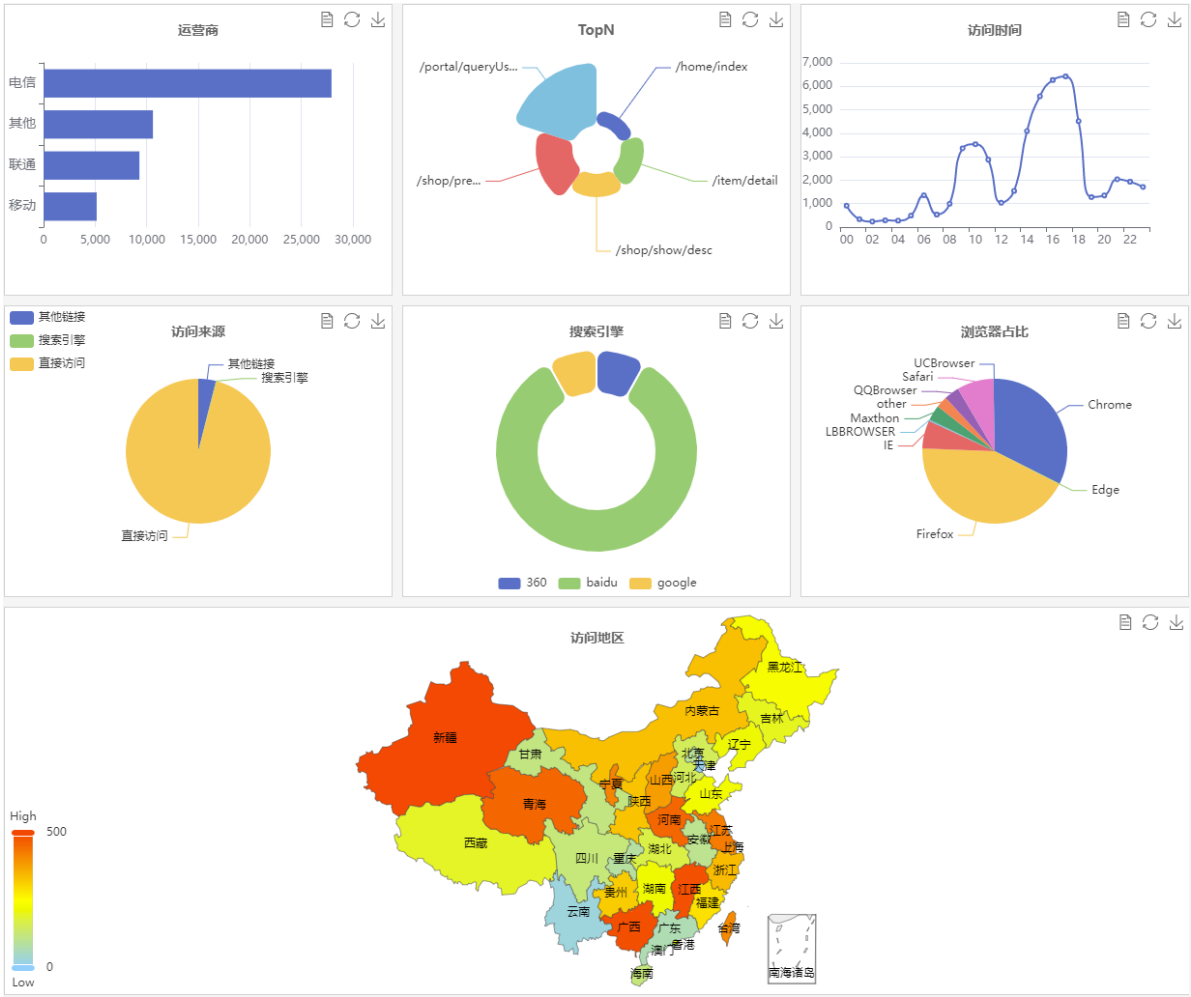
网站日志文件：access_index.log，数据格式如下：

```
1 120.26.64.126 - - [21/Aug/2017:23:59:03 +0800] "HEAD / HTTP/1.1" 301 0 "-"
"Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)"
2 112.126.73.56 - - [21/Aug/2017:23:59:03 +0800] "HEAD / HTTP/1.1" 301 0 "-"
"Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)"
```

ip地址库文件：ip.txt，数据格式如下：

```
0.0.0.0         0.255.255.255   IANA 保留地址
1.0.0.0         1.0.0.255       澳大利亚 亚太互联网络信息中心
1.0.1.0         1.0.3.255       福建省 电信
1.0.4.0         1.0.7.255       澳大利亚 墨尔本Goldenit有限公司
1.0.8.0         1.0.15.255      广东省 电信
1.0.16.0        1.0.31.255      日本 东京I2Ts Inc
1.0.32.0        1.0.63.255      广东省 电信
1.0.64.0        1.0.127.255     日本 広島県中区大手町Energia通信公司
1.0.128.0       1.0.255.255     泰国 TOTNET
1.1.0.0         1.1.0.255       福建省 电信
```

基于以上数据考虑得到以下效果：



2. 数据处理

2.1 环境准备

由于实验数据量比较大，运行SQL过程中可能会出现虚拟内存不足的问题

```
-----
Diagnostic Messages for this Task:
[2021-11-19 17:22:58.949]Container [pid=10843,containerID=container_1637303284659_0014_01_000007] is running 3706
90560B beyond the 'VIRTUAL' memory limit. Current usage: 376.2 MB of 1 GB physical memory used; 2.4 GB of 2.1 GB
virtual memory used. Killing container.
```

可以适当修改yarn-site.xml文件，调高内存限制，yarn.nodemanager.vmem-pmem-ratio参数代表虚拟内存与物理内存的比率，如果超过默认2.1倍的限制，进程会被关闭，可以调高yarn.nodemanager.vmem-pmem-ratio值，或将yarn.nodemanager.vmem-check-enabled=false，关闭虚拟内存检查

```
<configuration>
    <!-- Site specific YARN configuration properties -->
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.vmem-pmem-ratio</name>
        <value>4</value>
    </property>
</configuration>
```

调整后需要重新启动hadoop

```
cd /opt/server/hadoop-3.1.0/sbin
./stop-all.sh
./start-all.sh
```

创建原始表：

```
# -- 将数据当做一列放入表中，后续再使用sql进行分割处理
CREATE TABLE access_log_content (
    content STRING
);
CREATE TABLE ip_content (
    content STRING
);
```

导入数据：

```
load data local inpath '/opt/data/web.log' into table access_log_content;
load data local inpath '/opt/data/ip.txt' into table ip_content;
```

2.2 日志文件属性提取

```
DROP TABLE IF EXISTS access_log_tmp1;
CREATE TABLE access_log_tmp1 (
id BIGINT,
ip STRING,
ip_num BIGINT,       -- ip对应的十进制数
ip_1 BIGINT,         -- ip首位数字
access_time STRING, -- 访问时间
url STRING,          -- 访问链接
status STRING,       -- http状态码
traffic STRING,      -- 流量
referer STRING,      -- 来源
c_info STRING        -- 客户端信息
);
```

```
-- 使用正则表达式提取并向临时表插入数据
INSERT OVERWRITE TABLE access_log_tmp1
SELECT
CAST(split(content, ' ')[0] AS BIGINT) AS id,
split(content, ' ')[1] AS ip,
cast(split(split(content, ' ')[1], '\\.')[0] as int)  * 256 * 256 * 256 +
cast(split(split(content, ' ')[1], '\\.')[1] as int) * 256 * 256 +
cast(split(split(content, ' ')[1], '\\.')[2] as int) * 256 +
cast(split(split(content, ' ')[1], '\\.')[3] as int) AS ip_num,
cast(split(split(content, ' ')[1], '\\.')[0] as int) AS ip_1,
regexp_extract(content,'(\\[.*\\])',1) AS access_time,
split(regexp_extract(content,'(".*?")',1)," ")[1] AS url,
split(content,' ')[9] AS status,
split(content,' ')[10] AS traffic,
split(content,' ')[11]  AS referer,
regexp_extract(content,'\" (".*?")$',1) AS c_info
FROM access_log_content;
```

2.3 IP信息提取

```
-- 初始化IP表
DROP TABLE IF EXISTS cz_ip;
CREATE TABLE cz_ip (
ip_start BIGINT,    -- 起始ip对应的十进制
ip_start_1 STRING, -- 起始ip首位
ip_end BIGINT,      -- 结束ip对应的十进制
city STRING,        -- 城市
isp STRING          -- 运营商
);
```

```
-- 使用正则表达式提取并向IP表插入数据
INSERT OVERWRITE TABLE cz_ip
SELECT
CAST(split(split(content, '\\s+')[0],'\\.')[0] AS BIGINT) * 256 * 256 * 256 +
CAST(split(split(content, '\\s+')[0],'\\.')[1] AS BIGINT) * 256 * 256 +
CAST(split(split(content, '\\s+')[0],'\\.')[2] AS BIGINT) * 256 +
CAST(split(split(content, '\\s+')[0],'\\.')[3] AS BIGINT) AS ip_start,
split(split(content, '\\s+')[0],'\\.')[0] AS ip_start_1,
CAST(split(split(content, '\\s+')[1],'\\.')[0] AS BIGINT) * 256 * 256 * 256 +
CAST(split(split(content, '\\s+')[1],'\\.')[1] AS BIGINT) * 256 * 256 +
CAST(split(split(content, '\\s+')[1],'\\.')[2] AS BIGINT) * 256 +
CAST(split(split(content, '\\s+')[1],'\\.')[3] AS BIGINT) AS ip_end,
split(content, '\\s+')[2] AS city,
split(content, '\\s+')[3] AS isp
FROM ip_content;
```

2.4 信息合并（地域及运营商）

```
-- 初始化临时表
DROP TABLE IF EXISTS access_log_tmp2;
CREATE TABLE access_log_tmp2 (
id BIGINT,
ip STRING,
city STRING,
isp STRING,
access_time STRING,
url STRING,
status STRING,
traffic STRING,
referer STRING,
c_info STRING
);
-- 从ip表中查询城市和运营商信息
INSERT OVERWRITE TABLE access_log_tmp2
SELECT a.id, a.ip, b.city, b.isp, a.access_time, a.url, a.status, a.traffic,
a.referer, a.c_info
FROM access_log_tmp1 a JOIN
cz_ip b ON a.ip_1 = b.ip_start_1 AND a.ip_num >= b.ip_start AND a.ip_num <=
b.ip_end;
```

2.5 用户所在省和访问链接

```
-- 初始化临时表
DROP TABLE IF EXISTS access_log_tmp3;
CREATE TABLE access_log_tmp3 (
id BIGINT,
ip STRING,
province STRING,
city STRING,
isp STRING,
access_time STRING,
url STRING,
status STRING,
traffic STRING,
referer STRING,
c_info STRING
);
-- 提取省信息使用正则表达式提取访问链接的实际地址并向临时表插入数据
INSERT OVERWRITE TABLE access_log_tmp3
SELECT id,ip,
CASE WHEN INSTR(city, '省') > 0 THEN SUBSTR(city,1,INSTR(city,'省')-1)
WHEN INSTR(city,'内蒙古') > 0 THEN '内蒙古'
WHEN INSTR(city,'西藏') > 0 THEN '西藏'
WHEN INSTR(city,'广西') > 0 THEN '广西'
WHEN INSTR(city,'宁夏') > 0 THEN '宁夏'
WHEN INSTR(city,'新疆') > 0 THEN '新疆'
WHEN INSTR(city,'北京') > 0 THEN '北京'
WHEN INSTR(city,'上海') > 0 THEN '上海'
WHEN INSTR(city,'天津') > 0 THEN '天津'
WHEN INSTR(city,'重庆') > 0 THEN '重庆'
WHEN INSTR(city,'香港') > 0 THEN '香港'
WHEN INSTR(city,'澳门') > 0 THEN '澳门'
ELSE city end
AS province,
city,
isp,
access_time,
split(url,"\\?")[0] AS url
,status, traffic, referer, c_info
FROM access_log_tmp2;
```

2.6 访问时间和referer

```
-- 初始化临时表
DROP TABLE IF EXISTS access_log_tmp4;
CREATE TABLE access_log_tmp4 (
id BIGINT,
ip STRING,
province STRING,
city STRING,
isp STRING,
access_time STRING,
access_hour STRING,
url STRING,
```

```
status STRING,
traffic STRING,
referer STRING,
ref_type STRING,
c_info STRING
);
-- 使用正则表达式提取访问时间和来源分类并向临时表插入数据
INSERT OVERWRITE TABLE access_log_tmp4
SELECT id, ip, province, city, isp,
regexp_extract(access_time,":(\\d.*) ",1) AS access_time,
regexp_extract(access_time,":(\\d+):",1) AS access_hour
, url, status, traffic, referer
,
CASE WHEN
INSTR(referer, 'www.pdd.com') > 0 OR LENGTH(referer) < 5 THEN 'self'
WHEN INSTR(referer, 'www.google.com') > 0 THEN 'google'
WHEN INSTR(referer, 'www.baidu.com') > 0 THEN 'baidu'
WHEN INSTR(referer, 'www.bing.com') > 0 THEN 'bing'
WHEN INSTR(referer, 'www.so.com') > 0 THEN '360'
ELSE 'other'
END AS ref_type, c_info
FROM access_log_tmp3;
```

2.7 用户信息处理

```
-- 初始化访问日志表
DROP TABLE IF EXISTS access_log;
CREATE TABLE access_log (
id BIGINT,
ip STRING,
province STRING,
city STRING,
isp STRING,
access_time STRING,
access_hour STRING,
url STRING,
status STRING,
traffic STRING,
referer STRING,
ref_type STRING,
c_info STRING,
client_type STRING,
client_browser STRING
);
-- 使用正则表达式提取客户端信息中的操作系统和浏览器信息并向表插入数据
INSERT OVERWRITE TABLE access_log
SELECT id, ip, province, city, isp, access_time, access_hour, url, status,
traffic, referer, ref_type,c_info
,CASE
WHEN INSTR(c_info, 'iPhone;') > 0 THEN 'IOS'
WHEN INSTR(c_info, 'iPad;') > 0 THEN 'IOS'
WHEN INSTR(c_info, 'Mac OS X ') > 0 THEN 'OS X'
WHEN INSTR(c_info, 'X11;') > 0 THEN 'Linux'
WHEN INSTR(c_info, 'Android ') > 0 THEN 'Android'
WHEN INSTR(c_info, 'Windows NT ') > 0 THEN 'Windows'
```

```
ELSE 'other'
END AS client_type
,
CASE
WHEN INSTR(c_info, ' QQBrowser') > 0 THEN 'QQBrowser'
WHEN INSTR(c_info, ' UCBrowser') > 0 THEN 'UCBrowser'
WHEN INSTR(c_info, ' Edge') > 0 THEN 'Edge'
WHEN INSTR(c_info, ' LBBROWSER') > 0 THEN 'LBBROWSER'
WHEN INSTR(c_info, ' Maxthon') > 0 THEN 'Maxthon'
WHEN INSTR(c_info, ' Firefox') > 0 THEN 'Firefox'
WHEN INSTR(c_info, ' Chrome') > 0 THEN 'Chrome'
WHEN INSTR(c_info, ' Mac OS X') > 0
AND INSTR(c_info, ' Safari') > 0 THEN 'Safari'
WHEN INSTR(c_info, ' MSIE') > 0 THEN 'IE'
ELSE 'other'
END AS client_browser
FROM access_log_tmp4;
```

2.8 TopN数据采集

```
-- 初始化访问链接TopN表
DROP TABLE IF EXISTS access_log_url_top;
CREATE TABLE access_log_url_top (
url STRING,
times INT
);
INSERT OVERWRITE TABLE access_log_url_top
SELECT top.url, top.times
FROM
(
    SELECT url, COUNT(*) AS times
    FROM access_log
    GROUP BY url
) top
ORDER BY top.times DESC LIMIT 10;
```

获取每个访客的第一条访问日志，脚本如下： UV 独立访客 PV：页面浏览量

```
-- 初始化每个访客的第一个访问日志
DROP TABLE IF EXISTS access_log_first;
CREATE TABLE access_log_first (
id BIGINT,
ip STRING,
province STRING,
city STRING,
isp STRING,
access_time STRING,
access_hour STRING,
url STRING,
status STRING,
traffic STRING,
referer STRING,
ref_type STRING,
c_info STRING,
```

```
client_type STRING,
client_browser STRING
);
INSERT OVERWRITE TABLE access_log_first
SELECT a.id, a.ip, a.province, a.city, a.isp, a.access_time, a.access_hour,
a.url, a.status, a.traffic, a.referer, a.ref_type, a.c_info, a.client_type,
a.client_browser
FROM access_log a
JOIN
(
SELECT c.ip, MIN(c.id) AS id
FROM access_log c
GROUP BY c.ip,c.c_info
) b
ON a.ip = b.ip AND a.id = b.id;
```

2.9 IP黑名单处理

```
-- 初始化访问IP黑名单表
DROP TABLE IF EXISTS access_log_ip_black;
CREATE TABLE access_log_ip_black (
ip STRING,
times INT
);
INSERT OVERWRITE TABLE access_log_ip_black
SELECT ip, COUNT(1) AS times
FROM access_log WHERE status = '404'
GROUP BY ip
HAVING COUNT(*) > 10;
```

3. 数据分析

3.1 总访问人次（PV）：

```
select max(id) from access_log;
```

3.2 访问人数（UV）：

```
select count(id) from access_log_first;
```

3.3 独立IP：

```
select count(distinct ip) from access_log_first;
```

3.4 访问链接top10

```
select * from access_log_url_top;
```

3.5 各个时间访问人次：

```
select access_hour,count(id) from access_log group by access_hour;
```

3.6 访问来源占比：

```
select rt as name,count(rt) as value
from (
    select case ref_type
    when 'other' then '其他链接'
    when 'self' then '直接访问'
    else '搜索引擎' end as rt
    from t_web_access_log
) t group by rt
```

3.7 IP黑名单：

```
select * from access_log_ip_black;
```

3.8 地域分布：

```
select province,count(id) from access_log group by province;
```

3.9 操作系统占比：

```
select client_type,count(id) from access_log group by client_type;
```

3.10 浏览器占比：

```
select client_browser,count(id) from access_log group by client_browser;
```

3.11 网络运营商占比：

```
select t.isp, count(t.isp)from
(
select
case when
INSTR(isp,"移动") > 0 then "移动"
when INSTR(isp,"联通") > 0 then "联通"
when INSTR(isp,"电信") > 0 then "电信"
else "其他"
end as isp
from t_web_access_log
) t group by t.isp;
```

3.12 搜索引擎

```
select ref_type as name,count(ref_type) as value
from t_web_access_log
group by ref_type
having ref_type != 'self' and ref_type != 'other';
```

4. 数据迁移

4.1 创建MySql目标表

```
CREATE TABLE access_log
(
    id BIGINT,
    ip TEXT,
    province TEXT,
    city TEXT,
    isp TEXT,
    access_time TEXT,
    access_hour TEXT,
    url TEXT,
    status TEXT,
    traffic TEXT,
    referer TEXT,
    ref_type TEXT,
    c_info TEXT,
    client_type TEXT,
    client_browser TEXT
)DEFAULT CHARSET=utf8;
```

```
CREATE TABLE access_log_first
(
    id BIGINT,
    ip TEXT,
    province TEXT,
    city TEXT,
    isp TEXT,
    access_time TEXT,
    access_hour TEXT,
    url TEXT,
    status TEXT,
    traffic TEXT,
    referer TEXT,
    ref_type TEXT,
    c_info TEXT,
    client_type TEXT,
    client_browser TEXT
)DEFAULT CHARSET=utf8;
```

```
CREATE TABLE access_log_url_top
(
    url VARCHAR(200),
    times INT
)DEFAULT CHARSET=utf8;
```

```
CREATE TABLE access_log_ip_black
(
    ip VARCHAR(200),
    times INT
)DEFAULT CHARSET=utf8;
```

4.2 数据迁移

```
./sqoop export \
--connect "jdbc:mysql://node:3306/web_access?
useUnicode=true&characterEncoding=utf-8" \
--username root \
--password root \
--table access_log \
--export-dir /user/hive/warehouse/webaccess.db/access_log \
--input-fields-terminated-by '\001'
```

```
./sqoop export \
--connect "jdbc:mysql://node:3306/web_access?
useUnicode=true&characterEncoding=utf-8" \
--username root \
--password root \
--table access_log_first \
--export-dir /user/hive/warehouse/webaccess.db/access_log_first \
--input-fields-terminated-by '\001'
```

```
./sqoop export \
--connect "jdbc:mysql://node:3306/web_access?
useUnicode=true&characterEncoding=utf-8" \
--username root \
--password root \
--table access_log_url_top \
--export-dir /user/hive/warehouse/webaccess.db/access_log_url_top \
--input-fields-terminated-by '\001'
```

```
./sqoop export \
--connect "jdbc:mysql://node:3306/web_access?
useUnicode=true&characterEncoding=utf-8" \
--username root \
--password root \
--table access_log_ip_black \
--export-dir /user/hive/warehouse/webaccess.db/access_log_ip_black \
--input-fields-terminated-by '\001'
```