Flink sql

```
-- 执行sql查询      滚动窗口 10秒     计算10秒窗口内用户点击次数
SELECT TUMBLE_END(proctime, INTERVAL '10' SECOND) as processtime,
  userId,count(*) as pvcount
FROM Users
GROUP BY TUMBLE(proctime, INTERVAL '10' SECOND), userId
```

```
-- 数据库RDS结果表
CREATE TABLE source_ods_fact_log_track_action (
  account_id VARCHAR,  --用户ID
  client_ip VARCHAR,  --客户端IP
  client_info VARCHAR, --设备机型信息
  `action` VARCHAR, --页面跳转描述
  gpm VARCHAR, --埋点链路
  c_time BIGINT, --请求时间
  udata VARCHAR, --扩展信息，JSON格式
  `position` VARCHAR, --位置信息
  network VARCHAR, --网络使用情况
  p_dt VARCHAR  --时间分区天
) WITH (
  'connector.type' = 'kafka',
  'connector.version' = 'universal',
  'connector.topic' = 'topic_uv',
  'update-mode' = 'append',
  'connector.properties.zookeeper.connect' = '172.24.103.8:2181',
  'connector.properties.bootstrap.servers' = '172.24.103.8:9092',
  'connector.startup-mode' = 'latest-offset',
  'format.type' = 'json'
);

CREATE TABLE result_cps_total_summary_pvuv_min (
summary_date varchar, --统计日期
summary_min varchar, --统计分钟
pv bigint, --点击量
uv bigint, ---一天内同个访客多次访问仅计算一个UV
current_times varchar  --当前时间
-- primary key (summary_date, summary_min)
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'ali_pvuv',
  'connector.username' = 'root',
  'connector.password' = '123456'
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
);


select p_dt  as summary_date, --时间分区
    count (client_ip) as pv, --客户端的IP
```

```
    count (distinct client_ip) as uv, --客户端去重
    cast (max (FROM_UNIXTIME(c_time)) as TIMESTAMP) as c_time --请求的时间
from source_ods_fact_log_track_action
group by p_dt

INSERT into result_cps_total_summary_pvuv_min
select a.summary_date, --时间分区
  cast (DATE_FORMAT (c_time, 'HH:mm') as varchar) as summary_min, --取出小时分钟级别的时间
  a.pv,
  a.uv,
  cast (LOCALTIMESTAMP as varchar) as current_times --当前时间
from result_cps_total_summary_pvuv_min_01 AS a
```

```
-- 实时计算当天UV指标sql，这里使用最简单的group by agg，没有使用minibatch或窗口，在大数据量优化时最好使用后两种方式
insert into uv_index select log_date, ROW(count(distinct mid) as UV) from pageview group by log_date

```

```
use catalog my_hive;
-- build streaming database and tables;
create database stream_db;
use stream_db;
create table order_table (
    id long,
    amount double,
    user_id long,
    status string,
    ts timestamp,
    … -- 可能还有几十个字段
    ts_day string,
    ts_hour string
) with (
    'connector.type' = 'kafka',
    … -- Kafka table相关配置
);
-- build batch database and tables;
create database batch_db;
use batch_db;
create table order_table like stream_db.order_table (excluding options)
partitioned by (ts_day, ts_hour)
with (
    'connector.type' = 'hive',
    … -- Hive table相关配置
);

insert into [stream_db.|batch_db.]order_table select … from log_table;


```

```
-- stream 维表
use stream_db;
create table user_info (
    user_id long,
    age int,
    address,
    primary key(user_id)
```

```
) with (
    'connector.type' = 'jdbc',
    ...
);

-- 将离线数仓的维表导入实时数仓中
insert into user_info select * from batch_db.user_info;

-- 维表Join，SQL批流复用
insert into order_with_user_age select * from order_table join user_info for system_time as of order_table.proctime on user_info.user_id = user_info.user_id;

select age, avg(amount) from order_with_user_age group by age;
```

```
-- batch：计算完成后，一次性输出到mysql中，同key只有一个数据
-- streaming：mysql里面的数据不断更新，不断变化
insert into mysql_table select age, avg(amount) from order_with_user_age group by age;
-- batch: 同key只有一个数据，append即可
insert into hive_table select age, avg(amount) from order_with_user_age group by age;
-- streaming: kafka里面的数据不断append，并且多出一列，来表示这是upsert的消息，后续的Flink消费会自动做出机制来处理upsert
insert into kafka_table select age, avg(amount) from order_with_user_age group by age;
```

```
CREATE TABLE app_heartbeat_stream_source (
  `ip`                    VARCHAR,
  agent                   VARCHAR,
  roomid                  VARCHAR,
  userid                  VARCHAR,
  abytes                  VARCHAR,
  afcnt                   VARCHAR,
  adrop                   VARCHAR,
  afts                    VARCHAR,
  alat                    VARCHAR,
  vbytes                  VARCHAR,
  vfcnt                   VARCHAR,
  vdrop                   VARCHAR,
  vfts                    VARCHAR,
  vlat                    VARCHAR,
  ublock                  VARCHAR,
  dblock                  VARCHAR,
  region                  VARCHAR,
  stamp                   VARCHAR,
  app_ts AS TO_TIMESTAMP(stamp), --定义生成WATERMARK的字段，
  WATERMARK FOR app_ts AS app_ts - INTERVAL '10' SECOND --WATERMARK比数据时间线性增加10S
) WITH (
  'connector.type' = 'kafka',
  'connector.version' = 'universal',
  'connector.topic' = 'topic_live',
  'update-mode' = 'append',
  'connector.properties.zookeeper.connect' = '172.24.103.8:2181',
  'connector.properties.bootstrap.servers' = '172.24.103.8:9092',
  'connector.startup-mode' = 'latest-offset',
  'format.type' = 'json'
)

SELECT
  ip,
```

```
  agent,
  CAST(roomid AS BIGINT) as roomid,
  CAST(userid AS BIGINT) as userid,
  CAST(abytes AS BIGINT) as abytes,
  CAST(afcnt AS BIGINT) as afcnt,
  CAST(adrop AS BIGINT) as adrop,
  unix_timestamp(afts) as afts,
  CAST(alat AS BIGINT) as alat,
  CAST(vbytes AS BIGINT) as vbytes,
  CAST(vfcnt AS BIGINT) as vfcnt,
  CAST(vdrop AS BIGINT) as vdrop,
  unix_timestamp(vfts) as vfts,
  CAST(vlat AS BIGINT) as vlat,
  CAST(ublock AS BIGINT) as ublock,
  CAST(dblock AS BIGINT) as dblock,
  app_ts,
  region
FROM
  app_heartbeat_stream_source


SELECT
  CAST(TUMBLE_START(app_ts, INTERVAL '1' MINUTE) as VARCHAR) as app_ts,
  roomid,
  SUM(ublock) as ublock,
  SUM(dblock) as dblock,
  SUM(adrop) as adrop,
  SUM(vdrop) as vdrop,
  SUM(alat) as alat,
  SUM(vlat) as vlat
FROM
  view_app_heartbeat_stream
GROUP BY
  TUMBLE(app_ts,INTERVAL '1' MINUTE),roomid


CREATE TABLE output_1 (
  app_ts             VARCHAR,
  roomid             BIGINT,
  ublock             BIGINT,
  dblock             BIGINT,
  adrop              BIGINT,
  vdrop              BIGINT,
  alat               BIGINT,
  vlat               BIGINT
  --不支持 PRIMARY KEY (roomid)
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'live_output_1',
  'connector.username' = 'root',
  'connector.password' = '123456',
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
)

```

```
    
    
CREATE TABLE output_1 (
  app_ts             VARCHAR,
  roomid             BIGINT,
  ublock             BIGINT,
  dblock             BIGINT,
  adrop              BIGINT,
  vdrop              BIGINT,
  alat               BIGINT,
  vlat               BIGINT
  --不支持 PRIMARY KEY (roomid)
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'live_output_1',
  'connector.username' = 'root',
  'connector.password' = '123456',
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
)


INSERT INTO output_1
SELECT * FROM room_error_statistics_10min


SELECT
  CAST(TUMBLE_START(app_ts, INTERVAL '1' MINUTE) as VARCHAR) as app_ts,
  region,
  SUM(alat)/COUNT(alat) as alat,
  SUM(vlat)/COUNT(vlat) as vlat
FROM
  view_app_heartbeat_stream
GROUP BY
  TUMBLE(app_ts, INTERVAL '1' MINUTE), region


CREATE TABLE output_2 (
  app_ts VARCHAR,
  region VARCHAR,
  alat   DOUBLE,
  vlat   DOUBLE
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'live_output_2',
  'connector.username' = 'root',
  'connector.password' = '123456',
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
)


INSERT INTO output_2
```

```sql
SELECT * FROM region_lat_statistics_10min


SELECT
  CAST(TUMBLE_START(app_ts, INTERVAL '1' MINUTE) as VARCHAR) as app_ts,
  SUM(IF(ublock <> 0 OR dblock <> 0, 1, 0)) / CAST(COUNT(DISTINCT userid) AS DOUBLE) as
block_rate
FROM
  view_app_heartbeat_stream
GROUP BY
  TUMBLE(app_ts, INTERVAL '1' MINUTE)


CREATE TABLE output_3 (
  app_ts                VARCHAR,
  block_rate            DOUBLE
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'live_output_3',
  'connector.username' = 'root',
  'connector.password' = '123456',
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
)


INSERT INTO output_3
SELECT * FROM block_total_statistics_10min


SELECT
  CAST(TUMBLE_START(app_ts, INTERVAL '1' MINUTE) as VARCHAR) as app_ts,
  SUM(ublock+dblock) / CAST(COUNT(DISTINCT userid) AS DOUBLE) as block_peruser
FROM
  view_app_heartbeat_stream
GROUP BY
  TUMBLE(app_ts, INTERVAL '1' MINUTE)


CREATE TABLE output_4 (
  app_ts                    VARCHAR,
  block_peruser             DOUBLE
) WITH (
  'connector.type' = 'jdbc',
  'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
  'connector.table' = 'live_output_4',
  'connector.username' = 'root',
  'connector.password' = '123456',
  'connector.write.flush.max-rows' = '10',
  'connector.write.flush.interval' = '5s'
)


INSERT INTO output_4
SELECT * FROM block_peruser_statistics_10min
```

```
202
203
204 -- TopN
205 SELECT
206   CAST(TUMBLE_START(app_ts, INTERVAL '60' SECOND) as VARCHAR) as app_ts,
207   roomid as room_id,
208   COUNT(DISTINCT userid) as app_room_user_cnt
209 FROM
210   view_app_heartbeat_stream
211 GROUP BY
212   TUMBLE(app_ts, INTERVAL '60' SECOND), roomid
213
214
215 SELECT
216   app_ts,
217   room_id,
218   app_room_user_cnt,
219   ranking
220   --PRIMARY KEY (app_ts,room_id,ranking)
221 FROM
222 (
223   SELECT
224     app_ts,
225     room_id,
226     app_room_user_cnt,
227     ROW_NUMBER() OVER (PARTITION BY 1 ORDER BY app_room_user_cnt desc) AS ranking
228   FROM
229     view_app_room_visit_1min
230 ) WHERE ranking <= 10
231
232
233
234 CREATE TABLE output_7 (
235   app_ts                        VARCHAR,
236   roomid                        BIGINT,
237   app_room_user_cnt             BIGINT,
238   ranking                       BIGINT
239 ) WITH (
240   'connector.type' = 'jdbc',
241   'connector.url' = 'jdbc:mysql://172.24.103.3:3306/flink',
242   'connector.table' = 'live_output_7',
243   'connector.username' = 'root',
244   'connector.password' = '123456',
245   'connector.write.flush.max-rows' = '10',
246   'connector.write.flush.interval' = '5s'
247 )
248
249
250
251 INSERT INTO output_7
252 SELECT * FROM view_app_room_visit_top10
253
254
255
```