Designing Google Analysis backend

Wat is google analysis

Google Analytics is a web analytics service that tracks and reports website traffic, analyze in-depth detail about the visitors on the website. Each site will have unique tracking id.

Requirements

- Handle large write volume: Billions of write events per day.
- Handle large read/query volume: Millions of merchants wish to gain insight into their business. Read/Query patterns are time-series related metrics.
- Provide metrics to customers with at most one hour delay.
- Run with minimum downtime.
- Have the ability to reprocess historical data in case of bugs in the processing logic.

post event:

Refer to flow on page4

- **User Client/mobile apps browse**

  Access the website, embed tracking Code in the Webpage send async request for every user interaction.

- **Load balancer**

  The request reaches the load balancer (such as nginx),

  Load balancer distributing the request workload across multiple upstream services based on the strategy, Such as round robin.

- **API gateway service.**

  Microservice component

  The load balancer routes the request to API gateway service. Base on the prediction rule, redirect access service.

- **Access service(post):**

- Microservice component

  Extract the data from request and pre-process data involved :

  1: Extract and format the data structure, such as get tracking ID, hash page URL, browser type, IP

  2: Pack the message payload with data, drop message into kafka queue topics

  3: Apache Kafka is used for building real-time streaming data pipelines.

- **Offline subsystem**
  - Event log service (microservice)

    Event log service will listen to topics and call statistic service to save log Info.
  - Event statistic service: (microservice)

    log info into mysql database.

    Processed the offline statistics data and store into mysql database.

- **Real time system**

  This system will mainly process the data change very frequently. Such as PV , keyword search, counter.

  - Event process service:（microsevice）

    Event process service will listen to topic and continue process the data with more detail information

    Call event data service to persist the data

  - Event data service（microservice）

    Get the request from EPS and save into no-sql database like redis

get event:

metrics Data are come up with two section:

Real time data which update frequently

Offline data which at most one hour delay

The requests will be route from load balancer ->API gateway->access service(get)->data service(microservice)

Collect data from real time database and offline database and sent back to client.

Downtime solution:

1: All the services after the API gateway are automatically registered in the zookeeper like service.

2: Services will send heartbeat to zookeeper and let zookeeper maintain health service.

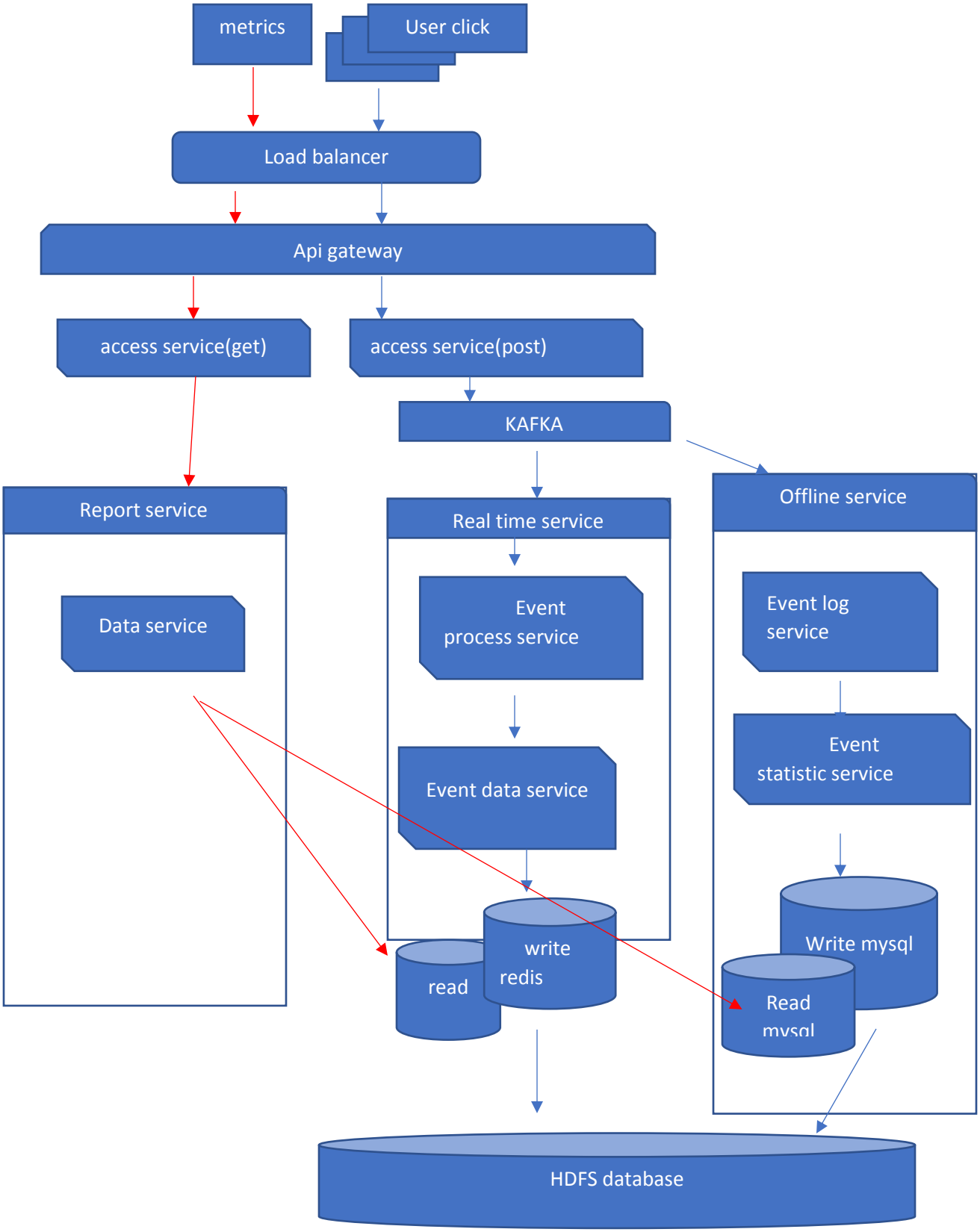
3: all services have the fallback rule, provide fallback response in case service is down.

Data replica and historical data archive:

1: All data in the redis and mysql will be stored and synchronized in multiple servers

2: Historical data from redis and mysql can be stored into big data database.

3: Separate the read and write db operation will improve the performance


metrics
User click
Load balancer
Api gateway
access service(get)
access service(post)
KAFKA
Report service
Data service
Real time service
Event
process service
Event data service
write
redis
read
Offline service
Event log
service
Event
statistic service
Write mysql
Read
mysql
HDFS database