**피드백**

**지원**

* 컨셉: 설문
  + 설문을 하게 되면 구글 폼 많이 사용함
  + 구글 폼하면 시트에 결과가 저장됨
  + 마케터 입장에서는 설문조사 하기 위해서 인프라 구축을 굳이 할 필요가 없음
  + 광고 같은거 활용하면 sns 활용해 google add 같은 기능 활용하면 수집할 수 있음? 광고 노출 쪽으로 쓸 수는 있지만….?
  + 수집은 다른 방향으로 많이 사용함
  + so 설문을 위해 인프라를 구축하는 것은 비용효율적이지 않음(비효율적임)
* 이 파이프라인을 쓰되 다른 방안으로 사용할 수 있지만

**민경**

* 아이디어는 현실적임
* 배민 리뷰이벤트 하는 이유가 있음. 사람들은 재확인하고 싶은 속성이 있음. 내가 보기에는 괜찮다고 느껴도 리뷰가 안달려있으면 사는걸 망설임. 리뷰있으면 구매 확류이 높음
* 아키텍처를 좀 더 구체화하면 좋을 것 같음
* 시작하는 관점에서 나쁘진 않음

**혜성**

* dms를 쓸 수 있음 하지만 지금 시나리오에는 조금 비현실적
* 실시간으로 하려면 s3로 스트림으로 꼽는게 나음
* 일반적으로 외부에 데이터가 있으면 가지고 올 때 dms를 사용함. 그런 관점에서 시도하는 것은 나쁘지 않은ㄴ데. 한 아키텍처에서 dms 사용하는 것은 조금 비현실적임. 비용적으로
* 실무에서는 저렇게 아키텍쳐를 짜지는 않을 것임
* s3에 실제 변경 될 때
* 일반적으로 oltp 성임. 이들의 저장구조는
  + 데이터 다른 관점에서
  + olap: 분석, 분석하기 위해서는 모든 데이터 사용해야함. 컬럼러 방식? 컬람단위로 저장됨. 컬럼단위로 저장하면 남여 두가지로 저장할 수 있음. 논리적 압축. 성별 저장시 남여 하고. 남자가 어디서부터 어디까지인지만 지정ㄹ하면 끝임. 레토드 단위면 남자 여자 모두 저장함. 저장하고 있는 저장 사이즈가 일일이 저장하는 OLTP는 큼.
  + oltp: transaction 쇼핑몰, 본인의 데이터만 보면 됨(남의 데이터 볼 필요x), 필요한 데이터는 내 데이터 한개임, 레코드 단위로 저장됨, 블록 단위로 저장되어 잇음. 바이트 단위로. 블록에는 레코드가 순서대로 저장됨. 인덱스 만들면 그 인덱스에 해당되는 값을 가지고 주소를 가지고 있는 오브젝트를 만듦. 이 값이 어느 주소에 있는지 만드는게 인덱스.
  + 사용관점이 다름 하나만 찝어 하는것, 다 가져와서 내가 원하는 데이터 만드는ㄴ 것은 저장 방식이 다름 이러한 이유로 olap
* S3는 객체 단위로 저장됨. S3 객체 하나에 여러 레코드가 있음. 이 중에 하나만 삭제하고 수정하면
* 해당하는 레코드가 이쓴 오브젝트를 찾아서 그 값을 빼거나 추가해서 수정한다음에, 그 값에 해당하는 오브젝트를 새로 만들어서 올리면 기본 오브젝트는 (수정되고 삭제된 변경 된) 삭제해야함
  + 이 개념으로 만들어 놓은게 ICEburg, DATALAKE
  + 이 개념으로ㅗ 메타데이터 관리해 변경된 오브젝트가 현재 사용하는 오브젝트와 매핑시켜 관리하고, ㅇ전 오브젝트로 인식하게 만드는 것4

⇒dms로 cnc로 밀어넣을 때 포멧이 다름. insert, delete 등 구분 못함. 오브젝트가 계속 추가될 것임. 앞에 플래그가 붙어 저장됨. so 누적된 작업에 insert, delte 구분 해야함.ㅏ

⇒ s3 저장구조가 카탈로그 만들 때 테이블 구조가 lakehouse 포멧을 아테나는 지원을함. 이렇게 바꿔놓으ㅕㅁㄴ 나중에 dms 통해 밀어넣을 때

⇒ 디테일이 빠졌음

**건호**

* 버퍼 사이즈가 잘못된 것 같음(시간 조정해보기)
* ec2→Kinesis Firehose → S3에서 버퍼 타임 및 사이즈 둘 중 하나를 충족해야 받아질텐데, 충족이 안돼서 시간이 많이 걸리는 것임
* Stream 쓰는 이유
  + kafka는 db라고 인식함. kafka는 데이터를 스트림으로 옮기는 목적이 있지만, 데이터를 저장함. 특정 기간 동안만 저장함. 저장하는 이유는 부하 조정, 버퍼 역할
  + 입력 받는 부분에서 확실히 데이터 입력 받으면 특정 기간동안은 데이터 확실하게 가지고 있음. 따라서 원할 때 데이터를 확실히 가져올 수 있음. (데이터 누락 방지 )
* 액세스 로그를 가공해서 좀 더 많으 데이터(클릭 id 값 등을 넣어서 함). 실제 액세스 로그로 볼 수 있는 데이터는 별로 없음. 이 정보를 가지고 할 수 있는게 많이 없음. ip 정도는 알 수는 있는데 누가 접근했는지 알수는 없음.
  + 실무에서는 세션, ip 정보 를 ip 테이블 매핑해서 저장해놧다가 나중에 반복적으로 들어온ㄴ ip는 동일하다고 생각하고, 만약 로그인 안되서 접근했을 때 이전 정보보고 이 ip로 봤을 때 이 유저일것이다라고 유추가능(unique 하게 들어오는 사용자 확인만 가능). 하지만 폰으로 들어오면 ip 달라지니까 다양한 ip로 들어올 수 있음
  + 어던 요구와 어떤 행동을 이해할 것인지 구체화 하기
* paginated 옵션이 있는데, 이게 자동으로 켜져있는데, 이것만 조심하기. 나머지는 비용 별로 나가지 않을 것임.
  + 리포트를 quicksight로 뽑아내는데

**공통과제**

* 주제를 누구거 하나를 선택하기 보다는 내용 버무리기
* 아키텍쳐 관점에서 **파이프라인 영역, 수집 영역**
  + 민경 혜성
    - 리뷰개수와 팬매율에 추가되기. 개수도 보고 점수도 보고
    - 혜썽, 민경은 두 개는 섞기 좋음
  + 지원
    - 현실적이지 않음
    - 소스로 엑셀 작업한 것을 로딩해서 파싱해서 아테나로 쿼리하는 데이터 일부로 쓴다든지
    - 그 데이터를 바로 s3로 밀어넣는게 좋을 거 같음
  + 건호
    - rds는 스트림으로 꽂는게 모범적임
    - 고객, 상품 정보 등은 db에서 가지고 와야함(실시간 필요없음. 배치)
    - 액세스 로그는 좋은 로그는 아님.
      * 액세스 로그와 비슷하게 스크립트를 쌓는 것
      * 일종의 스크립트 받아서 태그 들도 오브젝트인데, 이거 가져와서 스크롤까지 캐치할 수 있음
      * 스크립트 만들어놓은게거 호출할 수 있음
    - 자바 스크립트로 하나 만들어서 여기에 현재 이 사람의 세션 id 정보를 넘겨서 간단하게 상품id, 카테고리 id, 유저 id 넘겨서 서버에서 받아 저장하게
      * 작은 사이트를 보면 소스가 간단하고 오픈되어 있음. 필요한 정보를 입력하고 파라미터를 던지면..?
      * 호출되는 것 만들엇 액세스 로그 형태로 유사하게 남기는 것을 조금 더 디테일한 정보 남기게 하면 유의마한게 될 것임

**데이터**

1. 수집
   * 설문-엑셀로 떨어진다면, 수기든 자동이든 일단 s3로 가게끔
   * DMS→S3로 갈 때 DMS or SNAPSHOT
   * 실제 트랜젝션에 대한 ACCESS 로그: S3로 갈때 FIREHOSE(STREAM)
2. 데이터 마트 구성
   * glue crawler 사용해 datacatalog
     + 이때 glue crawler에서 meta partition 고민
     + 람다로 파티션해도 되고
   * data catalog
     + athena
3. visualizatio
   * quicksight
     + 대시보드를 어느 수준까지 만들 것인가
     + 대시보드를 잘 보여지게 해야 앞

| **수집** | **데이터 마트 구성** | **시각화** |
| --- | --- | --- |
| EXCEL | S3 |  |
| DB |  |  |
| -고객 정보 | S3 |  |
| transaction(access) |  |  |
| -로그 수집에 대한 고민해야함(보통 스크립트로함), 로그 수집 서버 있어야 함 | S3 |  |

* s3가 데이터 레이크임.
  + 하나의 버킷에 넣어도 되고 여러 버킷에 넣어도 됨. 하나의 account면
  + 버킷 나누는게 제일 심플함
  + 버킷 왜 나눴는지
    - 고객 정보이기 때문에 버킷 구분했다(실무에 가까움)
    - 이메일은 권한 있는 사람한테만 보이게끔 masking 한다든지(암호화)

대시보드는 필수임.

* 대시보드에 어떠한 사항 들어갈 것인지 먼저 정하기
* 대시보드 초안만들고 구체화하기

많은 것을 시도해봤으면 가산점

태스크로 나눠고 각 테스트를 구체화해서 현실적으로 할 수 있는 부분 등 구체화 해보기

다음 시간에 이렇게 wbs 구체화하고 rnr 나누기

수집보단 테스팅먼저 하고, 다하면 수집부분 진행하기

* 분석-시각화 중점으로 하고 수집은 가상으로 설정해서 하기

비용, 성능, 사용할 사람들의 skill set(이 사람이 쿼리에 익숙한지, python에 익숙한지)

구축에서 끝나는게 아니라 그들이 운영을 해야하기 때문에 그들의 skill set에 맞추는게 좋음

고객사 중에서 데이터 운영을 못하는 경우가 있음. 그러면 우리가 대행해주는 경우도 있음

인사이트가 나오면 새롭게 발견하고 싶은 욕구가 생기기 때문에 새로운 뷰를 요청해야햄. 이런 작업이 꾸준히 나타나야하는데 기술운영이 어려우면 대행 요청하기도 함

보안 안정화를 위해서는 엔드 포인트 최소화→ 노출되는 포인트가 작기 때문

고객들은 성능은 당연한 것이고, 비용은 적을 수록 좋음

* 배치성 관점에서는 최근 데이터들이 얼마나 더 빨리 동기화되어서 볼 수 있는가
* 대시보드 관점에서는 빠르게 데이터 볼 수 있어야 함
* quicksight를 실시간 처럼 보이려면 ..?
  + 실시간이라는게 30분 이내에만 볼 수 있으면 실시간 인거임. 고객사가 어떻게 생각하는거에 따름.

디테일은 떨어져도 얼마나 많은 관점에서 보고 시도해봤는지가 높은 평가를 받을 수 있는 척도가 될수도 있음

**개별 멘토링(개인 아키텍처)**

* **NatGW**
  + **nat gw 2개 두는 이유**는 하나 죽었을 때 한 쪽은 할 수 있게끔 하는 것임
  + 서버 세팅, 패치할 때 아웃바운드 트래픽 할 일
  + 가장 일반적으로는 결제 시스템으로 별도 통신할 때 사용함
  + 쓸 때 없이 띄워놨을 수도 있음
  + 실무에서는 pem key 사용, 배스턴 호스트 넣고
  + ? 지원 안할수도 있음
  + 세션 메니저는 나쁘지는 않지만 실제로 많이 사용하지는 않는 듯해

**건호**

* 실무에서는 깃에서 사용하는 기능들을 다 사용하려면 굳이 codecommit은 필요 없을 수도
* 깃허브 쓸 수 있으면 사용하고
* 소스코드를 외부영역에 빼지말라 라는 제약 조건이 있다면

**지원**

* t 타입은 잘 사용되지 않음
* 크레딧이 소모되는 순간 성능에 제한을 걸어버림. 서버가 죽었다 죽지 않았다
* ec2 인스턴스 타입에 따라… t타입은 그냥 싸게 만든 것임. 그 기준에 있어서 크레딧이 소모가 시작됨.
* 트래픽이 몰리는 상황에 적절하려면 어떤 ec2 인스턴스 타입 사용? ⇒ n타입 사용함
  + cpu 최적화⇒ db
  + r타입 ⇒ 데이터 많이 사용
* cloudfront는 캐시 되서 빨리 할 수 있도록 하는 것.
* 캐시에 모든 것을 다 때려놓지 않는 이유는 최고 빠르지만 비쌈
  + so cloudfront도 s3 사용 이유는 싸기 대문에 사용. 하지만 s3에 항상 접근 시 느림. 그러니까 비용 많이 접근하게 만드는 것임 사용 하지말라고. 그래서 cloudfront 만듦
  + 한 번 캐시된 영역은 빨리 캐시 되도록 최적화되어 있음

**민경**

* cloud agent를 깔아야 memory가 수집됨

**혜성**

* 모듈화 하지 않을 거면 테라폼 코드 main.tf에 하나로 합치기
* 모니터링이 cloudwatch가 아닌 x-ray 같은 다른 서비스 사용하는게 더 좋을 것 같음
  + 경보만 받는 것은 모니터링이 아님
* cloudfront는 기본적으로 shield가 standard로 제공됨
  + cloudfront가 중단되면 특정 리전, 특정 존이 맛이 감. 그러면 그 존에 있는 인스턴스에는 접근이 아예 안됨. 따라서 cloudfront는 장애가 나면 안됨. 기본적으로 DDoS 대응 할 것임.
* waf는 딱 이 서비스 공격을 막는 것. ddos는 수시로 계속적으로 일어나는 공격.
* cloudtrail 서비스
  + 어떤 사람이 사고쳤는지 확인하는 용도임. 리소스 수정 삭제 생성 등을 누가 했는지 확인하려고
  + cloudtrail을 athena로 연동해서 어떤 사람이 사고쳤는지 확인할 수는 있음
* Glue visual studio를 코드로 자동화(데이터 파이프라인 자동화 ⇒ 필수는 아님 )
  + dbt, workflow 관리하는 stepfunction, glue workflow를 활용해서 자동화하면 좋음
  + 하지만 핵심 위주로 하기 나중에 추가적으로 하기
  + 근데 요즘 workflow 사용하지 않는데는 없음

**공통**

* S3(datalake) - glue crawler -catalog - quicksight는 필수
* workflow도 추가적으로 할 수 있음
  + mwaa. airflow, stepfunction, glue workfow쓸 것인지 선택
  + DBT(서드 파티 솔루션)

wbs 나눠서 태스크 짜서, 롤 나눠서 각자 맡아서 하기