데이터의 종류

크게 업무에 필요한 데이터(**업무데이터**), 업무에 직접적으로 필요하지는 않지만 분석을 위해 추출해야 하는 데이터(**로그 데이터**)로 나눌 수 있음

업무데이터

- 서비스와 시스템을 운용하기 위한 목적으로 구축된 DB에 존재하는 데이터.
- 대부분 갱신형 데이터로, 데이터의 변경이 있을 때 새로운 데이터 삽입 대신 갱신
- **트랜잭션 데이터**
  - 서비스, 시스템을 통해 사용자의 행동을 기록한 데이터 ⇒ 구매, 리뷰, 게임플레이
  - 날짜, 시각, 회원 ID, 상품 ID, 수량, 가격 등
  - 회원의 성별, 주소 또는 상품의 카테고리, 이름 바로 추출 불가 ⇒ 마스터 데이터 필요!
- **마스터 데이터**
  - 서비스, 시스템이 정의하고 있는 데이터 ⇒ 회원 관련 정보, 상품 관련 정보
- 트랜잭션 데이터, 마스터 데이터를 결합하여 명확한 리포트를 만들 수 있음

로그 데이터

- 통계, 분석을 주 용도로 설계된 데이터
- 특정 태그를 포함하여 전송된 데이터
- 특정 행동을 서버 측에 출력한 데이터
- 이 책에서는 웹 서버 접근 로그뿐 아니라 전송 형식이나 파일 형식에 상관없이 모두 로그 데이터라고 부름
- 누적형 데이터로, 출력 시점의 정보를 저장하는 것이기에 데이터의 변동에 따라 데이터를 수정(갱신)할 필요가 없음

업무 데이터

특징

- **데이터의 정밀도가 높음**
  - 처리 도중 문제 발생⇒ 트랜잭션, 롤백 기능 사용하여 문제 제거 가능⇒ 데이터의 정합성 보증⇒ 정확한 값이 필요한 매출 관련 리포트 만들 때 유용
- **갱신형 데이터**
  - 사용자 탈퇴, 데이터 제거, 주문 취소, 주소 변경⇒ 데이터가 갱신되거나 제거되는 경우 있음 ⇒ 추출 시점에 따라 데이터가 바뀔 수 있음
- **다물 테이블 수가 많음**
  - 대부분 서비스 RDB 사용⇒ 데이터 정합성 유지하며 저장

5강. 하나의 값 조작하기

데이터를 가공해야 하는 이유

- 업무 데이터의 경우 DB에 코드 값만 저장하고, 코드의 의미를 다른 테이블에서 관리하는 경우 존재. ⇒ 매칭 필요
- 접근 로그는 하나의 문자열로 표현하는 경우 존재 ⇒ 형 변환, 분할 필요
- NULL 값

코드 값을 레이블로 변경하기

- 업무 데이터의 경우 DB에 코드 값만 저장하고, 코드의 의미를 다른 테이블에서 관리하는 경우 존재. ⇒ 매칭 필요

```
user_id   | register_date  | register_device
----------+----------------+-----------------
U001      | 2016-08-26     | 1
U002      | 2016-08-26     | 2
U003      | 2016-08-27     | 3
```

```
SELECT
  user_id
  , CASE
    WHEN register_device = 1 THEN '데스크톱'
    WHEN register_device = 2 THEN '스마트폰'
    WHEN register_device = 3 THEN '애플리케이션'
    ELSE ''
  END AS device_name
FROM mst_users
;
```

register_device 칼럼의 각 숫자를 레이블로 매칭시킴

> 조건 기반 값 결정; CASE
>
> CASE
> WHEN <조건> THEN <조건 만족 값>
> WHEN <조건> THEN <조건 만족 값>
> ELSE ' '
> END AS <값>

URL에서 요소 추출하기

5-2 URL에서 요소 추출하기

- 분석 현장에서는 로그 조건과 분석 요건을 제대로 검토하지 못하고
  - 최소한의 요건으로 `레퍼러(referer)`와 `페이지 URL`을 저장하는 경우 있음
- → 이후 URL 기반으로 추출 진행

```
CREATE TABLE ch3.access_log(
    stamp STRING
  , referrer STRING
  , url STRING
);
```

access_log 쿼리 공유 복사 스냅샷 삭제 내보내기

스키마 세부정보 PREVIEW 계보 데이터 프로필 데이터 품질

행	stamp	referrer	url
1	2016-08-26 12:02:00	http://www.other.com/path1/in..	http://www.example.com/vide..
2	2016-08-26 12:02:01	https://www.other.com/	http://www.example.com/book..
3	2016-08-26 12:02:01	http://www.other.net/path1/ind..	http://www.example.com/vide..

1. **레퍼러(referer)로 어떤 웹 페이지를 거쳐 넘어왔는지 판별하기**

- 어떤 웹페이지를 거쳐 넘어왔는지 판별 -> Referer
- 보통은 **호스트 단위**로 집계한다.
  - **페이지 단위**로 집계시, 밀도가 너무 작아 복잡하기 때문
- `Hive` 또는 `BigQuery`에는 URL을 다루는 함수 존재
  - 구현되지 않은 **미들웨어**에서는
  - **정규 표현식으로 호스트이름의 패턴**을 추출해야 함
- `Redshift`에서는 정규 표현식에서 괄호로 그룹화하는 기능이 없기 때문에
  - 정규식을 복잡하게 작성해야 함

```
SELECT
stamp, net.host(referrer) AS referrer_host
 From ch3.access_log;
```

행	stamp	referrer_host
1	2016-08-26 12:02:00	www.other.com
2	2016-08-26 12:02:01	www.other.com
3	2016-08-26 12:02:01	www.other.net

3. 문자열을 배열로 분리하기

- url 경로를 슬래시로 분할해 계층을 추출

```
select stamp, url
  ,split_part(substring(url from '//[^/]+([^?#]+)'), '/', 2) as path1
  ,split_part(substring(url from '//[^/]+([^?#]+)'), '/', 3) as path2
from access_log
```

- split_part 함수를 이용해 슬래시로 글자를 분리하고, 그 중에서 2번째와 3번째 글자를 각각 출력
- 파이썬에서 str.split('/')[2]와 동일한 기능

4. 날짜와 타임스탬프 다루기

- 현재 날짜와 타임스탬프 추출하기

```
select current_date as dt,
  current_timestamp as stamp
```

  - PostgreSQL 기준. MySQL은 Now()도 됨
- 지정한 값의 날짜/시각 데이터 추출하기

```
select cast('2016-01-30' as date) as dt
  , cast('2016-01-30 12:00:00' as timestamp) as stamp
```

  - cast(~~~ as datetype) 형식
- 날짜/시각에서 특정 필드 추출하기

```
select stamp
  , extract(year from stamp) as year
  , extract(month from stamp) as month
  , extract(day from stamp) as day
  , extract(hour from stamp) as hour

from
  (select cast('2016-01-30 12:00:00' as timestamp) as stamp) as t
```

  - extract(~~ from OO) 형식
  - 날짜 형식으로 뽑아내지 않더라도 문자열 형식으로도 가능 substr 함수