Lecture 10 - CSV와 SQL

개요

- 예제 데이터 둘러보기
- 예제 데이터 다운로드 및 CSV 이해하기
- csvkit 개요 및 간단한 실습
- SQL 및 csvsql 개요
- SQL 조인(join), SQL 집계(aggregation)
- 데이터 설계하기

예제 데이터 둘러보기 (5m)

예제 데이터: http://bit.ly/fastcamp-csv

세 개의 시트:

- customer: 고객 정보
- product: 상품 정보
- tx: 거래(transaction) 정보

예제 데이터 다운로드 및 CSV 이해하기 (15m)

파일 다운로드

```
curl http://s.g15e.com/customer.csv > customer.csv
curl http://s.g15e.com/product.csv > product.csv
curl http://s.g15e.com/tx.csv > tx.csv
```

잘 받아졌는지 확인하기

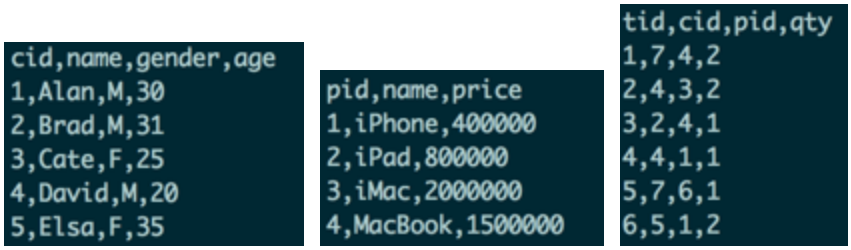
```
ls
```

파일 내용 출력해보기

```
cat customer.csv
cat product.csv
cat tx.csv
```

CSV?

- Comma-separated values 또는 Character-separated values (http://en.wikipedia.org/wiki/Comma-separated_values)



```
cid,name,gender,age
1,Alan,M,30
2,Brad,M,31
3,Cate,F,25
4,David,M,20
5,Elsa,F,35
```

```
pid,name,price
1,iPhone,400000
2,iPad,800000
3,iMac,2000000
4,MacBook,1500000
```

```
tid,cid,pid,qty
1,7,4,2
2,4,3,2
3,2,4,1
4,4,1,1
5,7,6,1
6,5,1,2
```

- 첫 줄에 헤더가 있거나 없거나
- 한 줄이 하나의 레코드
- 하나의 레코드는 쉼표 또는 탭 등으로 구분된 1개 이상의 칼럼(또는 필드)으로 구성

CSV는 왜 이리 널리쓰이나?

- 군더더기가 거의 없음
- 사람이 쉽게 읽고 쓸 수 있음
- 기계가 쉽게 읽고 쓸 수 있음
- 세상의 모든 데이터는 한 개 이상의 표로 표현할 수 있음 (http://en.wikipedia.org/wiki/Relational_model 참고)

csvkit 등 CSV 파일에 특화된 도구를 쓰면 CLI에서 좀 더 편하게 다룰 수 있음

csvkit 개요 및 간단한 실습 (15m)

설치하기

```
sudo apt-get install python3-pip
sudo pip3 install csvkit
```

무슨 차이인가?

```
cat customer.csv
```

```
cid,name,gender,age
1,Alan,M,30
2,Brad,M,31
3,Cate,F,25
4,David,M,20
5,Elsa,F,35
6,Fred,M,36
7,Glenn,M,36
8,Hellen,F,27
9,Ian,M,28
```

```
csvlook customer.csv
```

```
|------+--------+--------+------|
|  cid | name   | gender | age  |
|------+--------+--------+------|
|  1   | Alan   | M      | 30   |
|  2   | Brad   | M      | 31   |
|  3   | Cate   | F      | 25   |
|  4   | David  | M      | 20   |
|  5   | Elsa   | F      | 35   |
|  6   | Fred   | M      | 36   |
|  7   | Glenn  | M      | 36   |
|  8   | Hellen | F      | 27   |
|  9   | Ian    | M      | 28   |
|  10  | Jane   | F      | 29   |
|------+--------+--------+------|
```

cid, name, age 칼럼만 뽑아내기:

```
csvcut -c cid,name,age customer.csv | csvlook
```

간단한 통계량 보기:

```
csvstat customer.csv
```

(각 칼럼의 타입을 유추하여 타입에 따라 적절한 통계량을 보여줌)

정렬하기:

```
csvsort -c age customer.csv | csvlook
```

SQL 및 CSVSQL 개요 (15m)

SQL?

- **Structured Query Language**: RDBMS(Relational Database Management System)의 데이터를 조작하기 위한 선언적 언어(declarative language)
- RDBMS? 세상에서 가장 널리 쓰이는 데이터 저장/질의/관리 시스템. 관계대수(relation algebra)에 기반을 둔 이론적으로 탄탄한 접근법
- 선언적? “어떻게 하겠다"가 아니라 “무엇을 하겠다"를 적으면 **어떻게 할지는 기계가 알아서 수행**
    - 어떻게: 진공청소기로 청소를 한 번 하고 물걸레질을 한다.
    - 무엇을: 방에 먼지가 없어야 한다

csvsql:

- 보통 SQL은 RDBMS에 저장된 데이터에 질의(query)를 보내는 용도로 쓰임
- csvkit의 csvsql 명령을 쓰면 CSV 파일에 담긴 데이터에 SQL 명령으로 질의를 할 수 있음
- 주의: 데이터 양이 많은 경우에는 추천하지 않음. **빠르게 둘러보기 위한 용도로 적합**

csvsql 예시:

```
csvsql customer.csv --query "SELECT age, gender FROM customer;" | csvlook
```

너무 기니까 함수로 만들기:

```
query() { csvsql customer.csv product.csv tx.csv --query "$1" | csvlook; }
```

테스트:

```
query "SELECT age, gender FROM customer;"
```

SQL 질의 기초:

```
-- 모두 가져오기
SELECT * FROM customer;

-- 특정 칼럼만 가져오기
SELECT name, age FROM customer;

-- 나이가 30살 초과인 사람만
SELECT name, age FROM customer WHERE age > 30;

-- 나이가 30살 초과인 여성만
SELECT name, age FROM customer WHERE gender='F' and age > 30;

-- 여성만, 나이순으로
SELECT name, age FROM customer WHERE gender='F' ORDER BY age;

-- 여성만, 이름역순으로(DESCending) 정렬하여 첫 세 줄만(LIMIT 3)
```

```
SELECT name, age FROM customer WHERE gender='F' ORDER BY name DESC LIMIT 3;
```

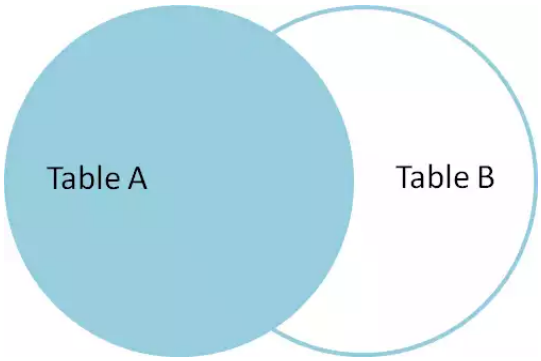
휴식 (10m)

SQL 조인 (20m)

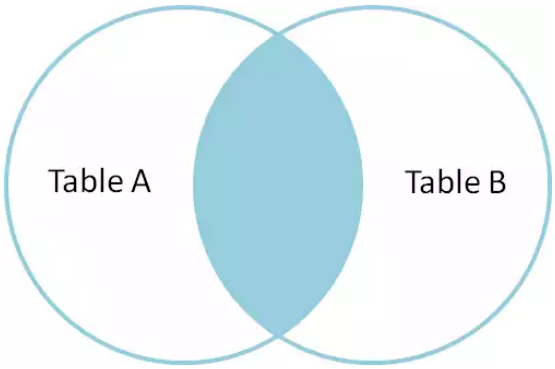
```
-- tx 테이블, customer 테이블 조인
SELECT t.tid, t.qty, t.cid, c.name
FROM tx t
LEFT OUTER JOIN customer c ON (t.cid=c.cid);

-- tx 테이블, customer 테이블, product 테이블 조인
SELECT t.tid, t.qty, t.cid, c.name, t.pid, p.price
FROM tx t
LEFT OUTER JOIN customer c ON (t.cid=c.cid)
LEFT OUTER JOIN product p ON (t.pid=p.pid);
```

Left Outer Join:



Inner Join:




```
-- tx 테이블, customer 테이블, product 테이블 조인
SELECT t.tid, t.qty, t.cid, c.name, t.pid, p.price
```

```
FROM tx t
INNER JOIN customer c ON (t.cid=c.cid)
INNER JOIN product p ON (t.pid=p.pid);
```

SQL 집계 (5m)

```
-- 성별 회원수
SELECT gender, COUNT(*) FROM customer GROUP BY gender;

-- 성별 연령대 평균, 최소, 최대값
SELECT gender, AVG(age), MIN(age), MAX(age) FROM customer GROUP by gender;
```

모두 합치기 (5m)

```
-- 25세 초과인 사람들이 가장 많이 구매한 상품은?
SELECT t.pid, p.name, SUM(t.qty), SUM(p.price)
FROM tx t
INNER JOIN customer c ON (t.cid=c.cid)
INNER JOIN product p ON (t.pid=p.pid)
WHERE age > 25
GROUP BY t.pid
ORDER BY SUM(t.qty) DESC;
```

데이터 설계하기 (20m)

원본:

order id	name	gender	age	product	price	qty
1	Alan	M	30	iPad	800000	1
2	Cate	F	25	iPhone	400000	1
3	Brad	M	31	iMac	2000000	1
4	Alan	M	30	iPhone, iPad	400000, 800000	2,1

문제들:

- 갱신: Alan이 개명을 하거나 성전환 수술을 하거나 나이를 먹으면? 모두 찾아서 고쳐야 함
- 삽입: 새로운 고객 David가 가입. 하지만 아직 아무 주문도 하지 않았다면?
- 삭제: Brad가 주문 취소를 하였다. Brad 정보도 모두 삭제하나?

개선 #1 - 하나의 칼럼에는 하나의 값만 담기

order id	name	gender	age	product	price	qty
1	Alan	M	30	iPad	800000	1
2	Cate	F	25	iPhone	400000	1
3	Brad	M	31	iMac	2000000	1
4	Alan	M	30	iPhone	400000	2
4	Alan	M	30	iPad	800000	1

개선 #2 - primary key에 함수종속 (functional dependency)

order id	cust id	product	qty
1	1	1	1
2	2	2	1
3	3	3	1
4	1	2	2
4	1	1	1
cust id	**name**	**gender**	**age**
1	Alan	M	30
2	Cate	F	25
3	Brad	M	31
product id	**product**	**price**	
1	iPad	800000	
2	iPhone	400000	
3	iMac	2000000	

개선 #3 - 외부의 변화에 종속적이지 않게

cust id	name	gender	birth year
1	Alan	M	1986
2	Cate	F	1991
3	Brad	M	1985