

2020년 한양대학교

# 소프트웨어융합 종합학술대회

---

[분야: 인턴십/캡스톤]

## 제목: The Data Civilizer System

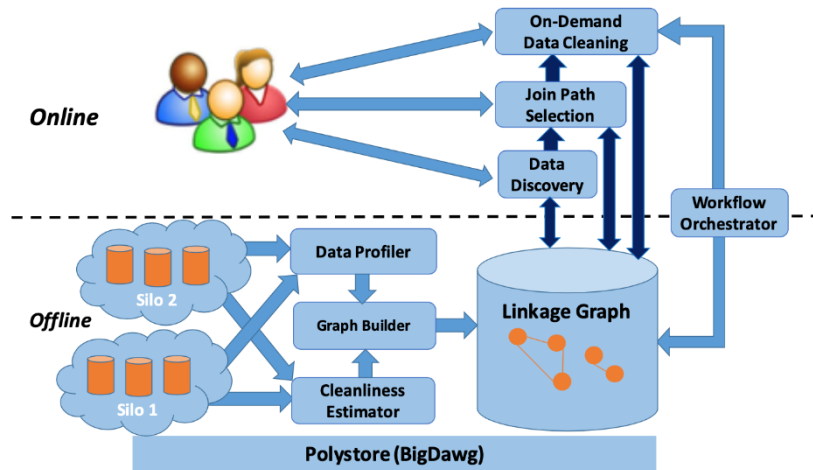
강은지(18학번), 석예림(18학번), 박서연(18학번)

2020. 12. 01

한양대학교 SW중심대학사업단



# The Data Civilizer System



end-to-end 빅데이터 관리 시스템

## Linkage Graph

데이터에 대한  
링크 그래프 구축

## Discovery

링크 그래프를 활용하여  
사용자 작업에 관련한 데이터 식별



- 링크 그래프를 사용하여 join path 찾기
- 실제 query 실행에서 polystore DBMS 사용
- 데이터 클리닝 작업을 query에 통합



# Data preprocessing

## - 결측치 제거

```
def cleanliness(df):  
    for x in df.columns:  
        df[x] = df[x].apply(lambda x: None if x == "null" else x)  
        if 'int64' == df[x].dtype:  
            lis = list(df[x])  
            for i in range(len(lis)):  
                df = validate(x, df, lis[i])  
            IQR(df, x, lis)  
    return df
```

## - Outlier 제거

```
def validate(x, df, date):  
    try:  
        datetime.datetime.strptime(str(date), '%Y%m%d')  
    except ValueError:  
        df[x] = df[x].replace(date, np.nan)  
    return df
```

## - 날짜 데이터 유효성 검사

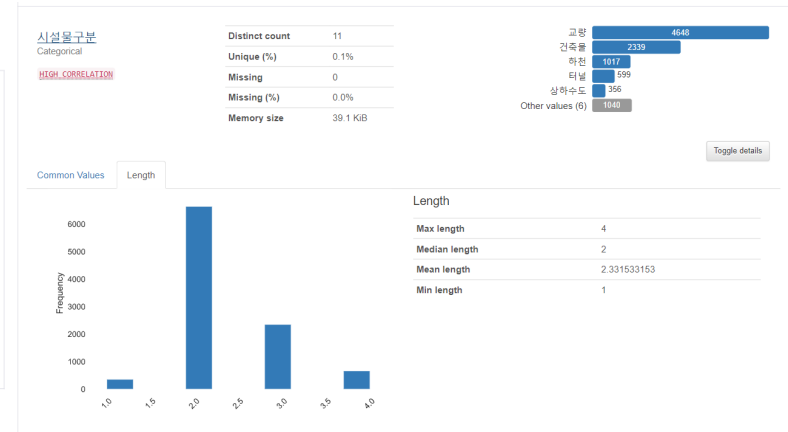
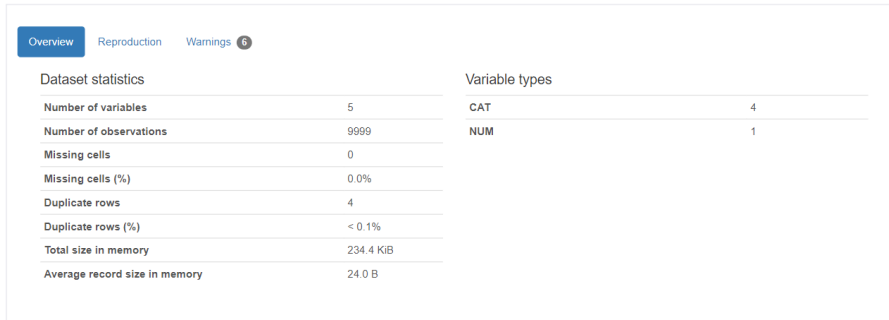
```
def IQR(df, x, lis):  
    Q1 = df.quantile(.25)  
    Q3 = df.quantile(.75)  
    IQR = Q3 - Q1  
    for i in range(len(lis)):  
        if ((df[x] < (Q1 - 1.5 * IQR)) | (df[x] > (Q3 + 1.5 * IQR))):  
            df[x] = df[x].replace(lis[i], np.nan)  
    return df
```



# Data Profiling : pipeline 구조

- pandas\_profiling 라이브러리를 사용하여 profiling 진행 및 메타 데이터 추출

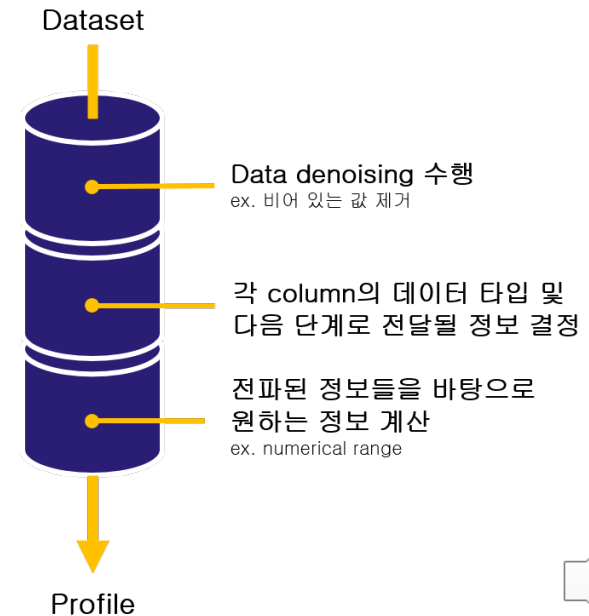
## Overview



- 칼럼 간 유사성 및 inclusion dependency 계산 구현

```
def col_similarity(table1, table2):
    res = {}
    col_sim = []
    ind = []
    for x in table1:
        for z in table2:
            top = 0
            dict(x[3])
            dict(z[3])
            top_key = x[3].keys()
            bottom_key = z[3].keys()
            for y in top_key:
                if y in bottom_key:
                    top += min(int(x[3][y]), int(z[3][y]))
            bottom = int(x[1]) + int(z[1]) - top # column1 + column2 - (col1과 col2의 교집합)
            if (top != 0 and bottom != 0):
                col_sim += [[x[0], z[0], float(top / bottom)]]
    if (len(col_sim) != 0):
        res['column_similarity'] = col_sim
    return res

def inclusion_dependency(table1, table2):
    res = {}
    ind = []
    for x in table1:
        for z in table2:
            dict(x[3])
            dict(z[3])
            top_key = x[3].keys()
            bottom_key = z[3].keys()
            if len(set(top_key) & set(bottom_key)) != 0:
                IND = float(
                    len(set(top_key) & set(bottom_key)) / len(set(top_key))
                ) # col_name의 교집합 / x의 집합 == 1일 경우 IND 설정
                if IND == 1:
                    ind += [[x[0], z[0], 'IND']]
    if (len(ind) != 0):
        res['inclusion_dependency'] = ind
    return res
```



# Data Update

- 데이터베이스 쿼리로그를 읽어와서 업데이트가 된 테이블에 대하여 프로파일링 재진행
- 기존 프로파일링과 비교하여 바뀐 사항 업데이트

## DB의 query log

```
SELECT * FROM mysql.general_log ORDER BY event_time DESC LIMIT 5;
```

event_time	user_host	thread_id	server_id	command_type	argument
2020-09-06 00:11:43.867121	root[root] @ localhost [::1]	8	1	Query	SELECT * from mysql.general_log
2020-09-06 00:12:33.792293	[root] @ localhost [::1]	9	1	Connect	root@localhost as anonymous on capstone
2020-09-06 00:12:33.795262	root[root] @ localhost [::1]	9	1	Query	SET AUTOCOMMIT = 0
2020-09-06 00:12:33.795886	root[root] @ localhost [::1]	9	1	Query	SHOW TABLES
2020-09-06 00:12:33.798075	root[root] @ localhost [::1]	9	1	Query	DROP TABLE test1

## 저장되어 있는 마지막 쿼리문의 시간보다 최근의 쿼리문만 모두 불러오기

```
SELECT *  
FROM mysql.general_log  
WHERE event_time > '2020-09-06 00:14:55'  
ORDER BY event_time DESC;
```

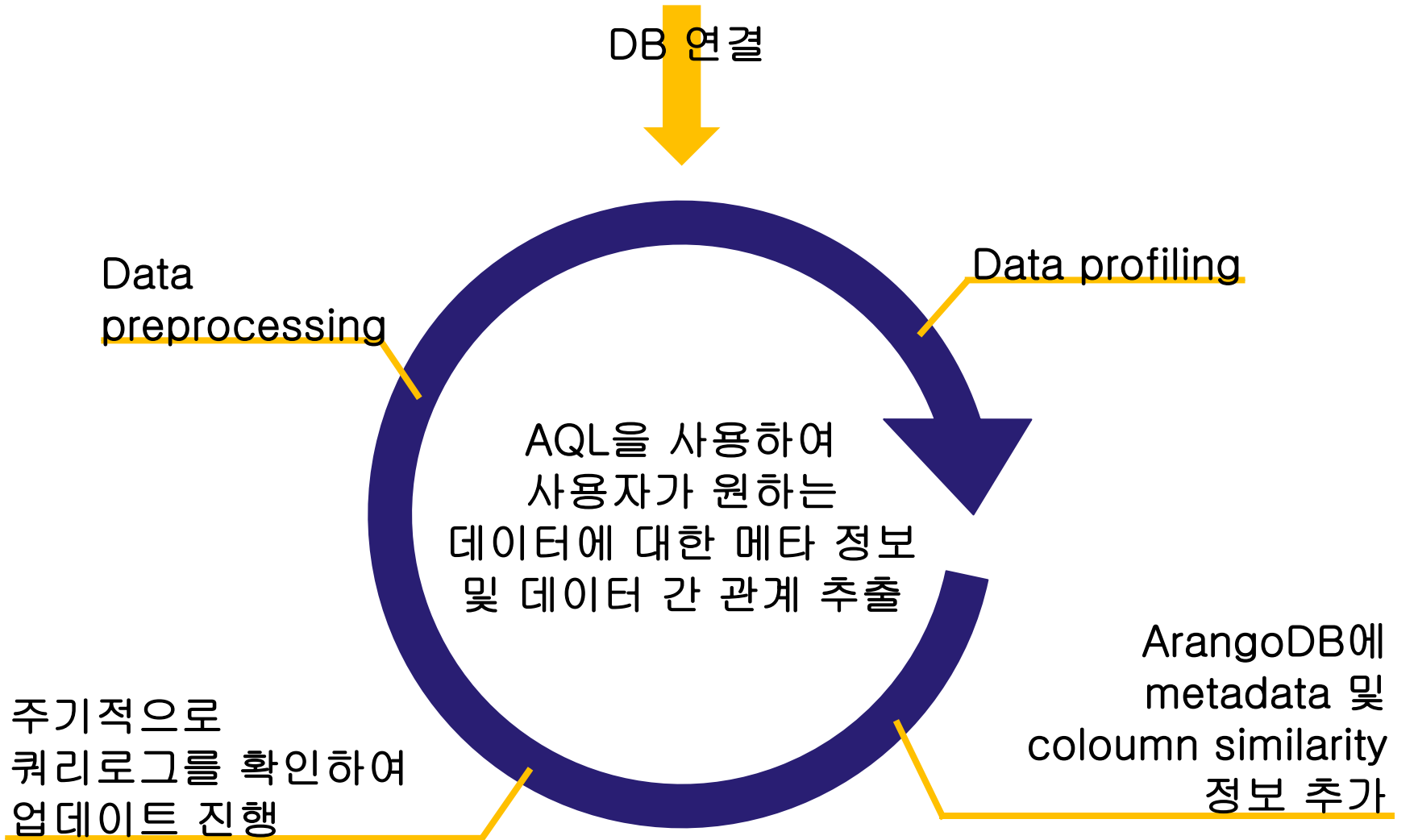
```
INSERT INTO test3 (Name, Classification, Type)  
VALUES ('07WK35D12900', '절토사면', '도로사면')
```

불러온 쿼리문에서 테이블명 추출  
-> profile 진행



# Data Civilizer System Structure

---



# Data Civilizer 구현 결과

- ArangoDB에 테이블의 메타데이터 입력

```
_id: test1/Classification
_rev: _bKpAja---
_key: Classification
```

Select a node...

object {6}

- name: Classification
- value\_counts: {'건축물': '9442', '교량': '4916', '하천': '1082', '터': '1082'}
- type: Variable.TYPE\_CAT
- max\_length: 4
- mean\_length: 2.595479968040178
- min\_length: 1

```
_id: test2/End_date
_rev: _bLlqUG---
_key: End_date
```

Select a node...

object {7}

name: End\_date

value\_counts: {'20191231': '1861', '20190630': '962', '20191130': '299', '20190531': '188', '20191228': '179', '20190930': '178', '20191227': '151', '20191130': '140', '20191225': '140', '20191224': '138', '20191223': '137', '20190430': '136', '20190228': '132', '20190629': '132', '20190930': '132', '20190430': '128', '20191101': '128', '20190831': '128', '20190531': '125', '20190528': '125', '20190604': '122', '20190331': '122', '20190330': '122', '20191201': '121', '20190322': '119', '20190311': '119', '20190415': '119', '20191213': '119', '20191211': '116', '20190719': '116', '20190807': '116', '20191010': '116', '20190810': '114', '20191114': '114', '20190712': '114', '20190613': '114', '20190414': '112', '20191109': '112', '20190402': '112', '20191024': '112', '20191209': '111', '20190411': '111', '20190921': '111', '20190905': '111', '20190618': '110', '20190616': '110', '20190427': '110', '20191104': '110', '20190623': '109', '20191215': '109', '20190830': '109', '20191211': '109', '20190603': '107', '20190325': '107', '20190410': '107', '20190821': '107', '20190918': '106', '20190428': '106', '20190822': '106', '20191111': '106', '20190811': '106', '20190405': '106', '20190805': '105', '20190803': '105', '20190810': '105', '20190814': '105', '20190222': '105', '20190811': '105', '20190805': '104', '20190802': '104', '20191007': '104', '20190808': '104', '20190807': '103', '20190817': '103', '20190821': '103', '20190827': '103', '20190805': '103', '20190818': '102', '20190706': '102', '20190602': '102', '20190827': '102', '20191012': '102', '20190824': '102', '20190908': '102', '20190827': '102', '20191014': '101', '20190809': '101', '20190420': '101', '20190822': '101', '20190826': '101', '20191008': '101', '20190813': '101', '20190728': '101', '20190726': '101', '20191006': '101', '20190528': '101', '20190711': '101'}

type: Variable.TYPE\_NUM

max: 20191231

min: 20190101

mean: 20190807.766920153

range: 1130

```
_id: test1/test1
_rev: _bKpAja---
_key: test1
```

Select a node...

object {3}

n: 17522

missing: 1

columns [5]

- 0: Name
- 1: Classification
- 2: Location
- 3: Completion\_Date
- 4: Type

- 칼럼간 유사성 및 inclusion dependency 데이터 입력

```
_id: relation/coltest3.NameAndtest1.Name
_rev: _bKpAleG---
_key: coltest3.NameAndtest1.Name
```

Select a node...

object {2}

- type: column\_similarity
- value: 0.10333844237953826

```
_id: relation/INDtest3.ClassificationAndtest2.Classification
_rev: _bLlqUG---
_key: INDtest3.ClassificationAndtest2.Classification
```

Select a node...

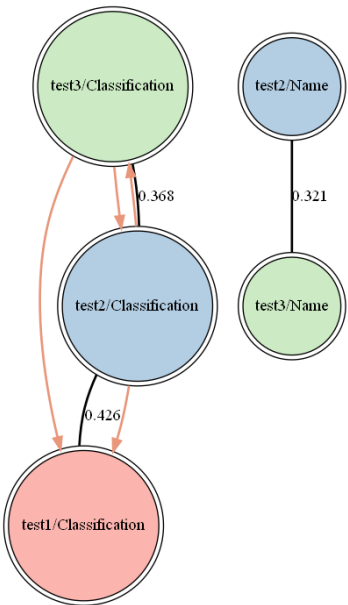
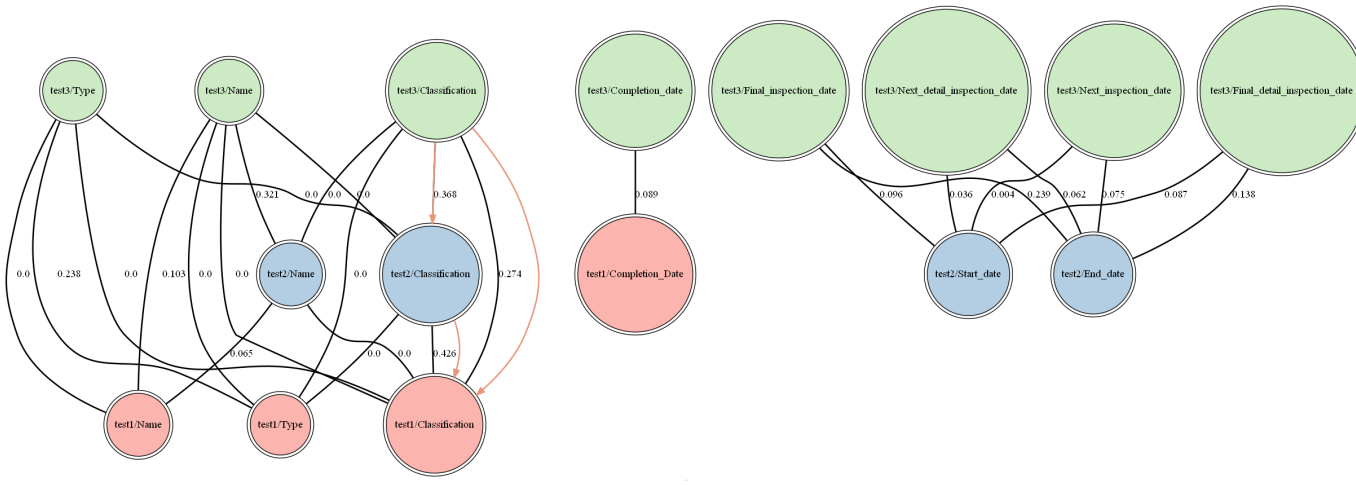
object {2}

- type: inclusion\_dependency
- value: 1



# Data Civilizer 구현 결과

*FOR x IN relation RETURN x*



*FOR x IN relation FILTER x.value > 0.3 RETURN x*

