

TinyIoT

Berkeley DB for TinyIoT

박민지

Berkeley DB

이번 주 진행사항

- I DB 저장 방식에 대한 고민
- II Type별 ACP 함수 성능 비교
- III CNT 구조체 수정

▶ 저장 방식 별 DB구조 비교

| Type1 | Type2 | Type3 |
|---|--|---|
| key에 속성의 이름을 저장하고, value에 속성값을 저장하는 구조 | key로 id를 사용하고, value에 속성 값을 저장하는 구조 | key로 id를 사용하고, value에 구분자 를 이용해 속성값을 나열하는 구조 |
| | | 최대 문자열 길이 임시로 200으로 설정 |

Key : Value

```
aei : TAE1
aei : TAE3
aei : TAE2
api : tinyProject1
api : tinyProject3
api : tinyProject2
ct : 20220513T083900
ct : 20220513T083900
ct : 20220513T083900
et : 20240513T083900
et : 20240513T083900
et : 20240513T083900
lt : 20220513T083900
lt : 20220513T083900
lt : 20220513T083900
pi : 5-20191210093452845
pi : 5-20191210093452845
pi : 5-20191210093452845
ri : TAE1
ri : TAE3
ri : TAE2
rn : Sensor1
rn : Sensor3
rn : Sensor2
rr : true
rr : true
rr : true
ty : 2
ty : 2
ty : 2
```

Key : Value

```
TAE1 : TAE1
TAE1 : tinyProject1
TAE1 : 20220513T083900
TAE1 : 20240513T083900
TAE1 : 20220513T083900
TAE1 : 5-20191210093452845
TAE1 : Sensor1
TAE1 : true
TAE1 : 2
TAE2 : TAE2
TAE2 : tinyProject2
TAE2 : 20210513T083900
TAE2 : 20230513T083900
TAE2 : 20210513T083900
TAE2 : 5-20191210093452845
TAE2 : Sensor2
TAE2 : true
TAE2 : 2
TAE3 : TAE3
TAE3 : tinyProject3
TAE3 : 20200513T083900
TAE3 : 20220513T083900
TAE3 : 20200513T083900
TAE3 : 5-20191210093452845
TAE3 : Sensor3
TAE3 : true
TAE3 : 2
```

Key : Value

```
TAE1 : Sensor1,
      5-20191210093452845,
      20220513T083900,
      20240513T083900,
      20220513T083900,
      true,
      2,
      TAE1,
      tinyProject1

TAE2 : Sensor2,
      5-20191210093452845,
      20220513T083900,
      20240513T083900,
      20220513T083900,
      true,
      2,
      TAE2,
      tinyProject2

TAE3 : Sensor3,
      5-20191210093452845,
      20220513T083900,
      20240513T083900,
      20220513T083900,
      true,
      2,
      TAE3,
      tinyProject3
```

▶ ACP_Store 저장 방식 별 DB구조 비교

Type1

Key : Value

```
ct : 20191210T093452
et : 20211210T093452
lt : 20191210T093452
pi : pipipi
pv_acop : 63
pv_acor : CAE1
pvs_acop : 63
pvs_acor : SM
ri : 5-20191210093452845
rn : acp1
ty : 5
```

Type2

Key : Value

```
5-20191210093452845 : acp1
5-20191210093452845 : pipipi
5-20191210093452845 :
5-20191210093452845 : 20191210T093452
5-20191210093452845 : 20191210T093452
5-20191210093452845 : 20211210T093452
5-20191210093452845 : CAE1
5-20191210093452845 : 63
5-20191210093452845 : SM
5-20191210093452845 : 63
```

Type3

Key : Value

```
5-20191210093452845 : acp1,pipipi,20191210T093452,20191210T093452,20211210T093452,5,CAE1,2,SM,63
```

▶ Get_ACP 저장 방식 별 실행 시간 비교(서버)

Type1

```
Add Child
[P] TinyIoT
[C] acp_test1...OK
프로그램 수행 시간 :0.000481

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test2
[ri] 1-20221013T0502196457...OK

Add Child
[P] TinyIoT
[C] acp_test2...OK
프로그램 수행 시간 :0.000391

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test3
[ri] 1-20221013T0502226907...OK

Add Child
[P] TinyIoT
[C] acp_test3...OK
프로그램 수행 시간 :0.000356

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test4
[ri] 1-20221013T0502248688...OK

Add Child
[P] TinyIoT
[C] acp_test4...OK
프로그램 수행 시간 :0.000479
```

Type2

```
Add Child
[P] TinyIoT
[C] acp_test2...OK
프로그램 수행 시간 :0.000387

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test3
[ri] 1-20221013T0505131414...OK

Add Child
[P] TinyIoT
[C] acp_test3...OK
프로그램 수행 시간 :0.000438

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test4
[ri] 1-20221013T0505159057...OK

Add Child
[P] TinyIoT
[C] acp_test4...OK
프로그램 수행 시간 :0.000355

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test5
[ri] 1-20221013T0505208354...OK

Add Child
[P] TinyIoT
[C] acp_test5...OK
프로그램 수행 시간 :0.000484
```

Type3

```
Add Child
[P] TinyIoT
[C] acp_test3...OK
프로그램 수행 시간 :0.000396

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test4
[ri] 1-20221013T0507091488...OK

Add Child
[P] TinyIoT
[C] acp_test4...OK
프로그램 수행 시간 :0.000458

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test5
[ri] 1-20221013T0507127781...OK

Add Child
[P] TinyIoT
[C] acp_test5...OK
프로그램 수행 시간 :0.000396

+ [POST] /TinyIoT
Parse_URI /TinyIoT...OK
Create ACP

Create Tree Node
[rn] acp_test6
[ri] 1-20221013T0507291222...OK

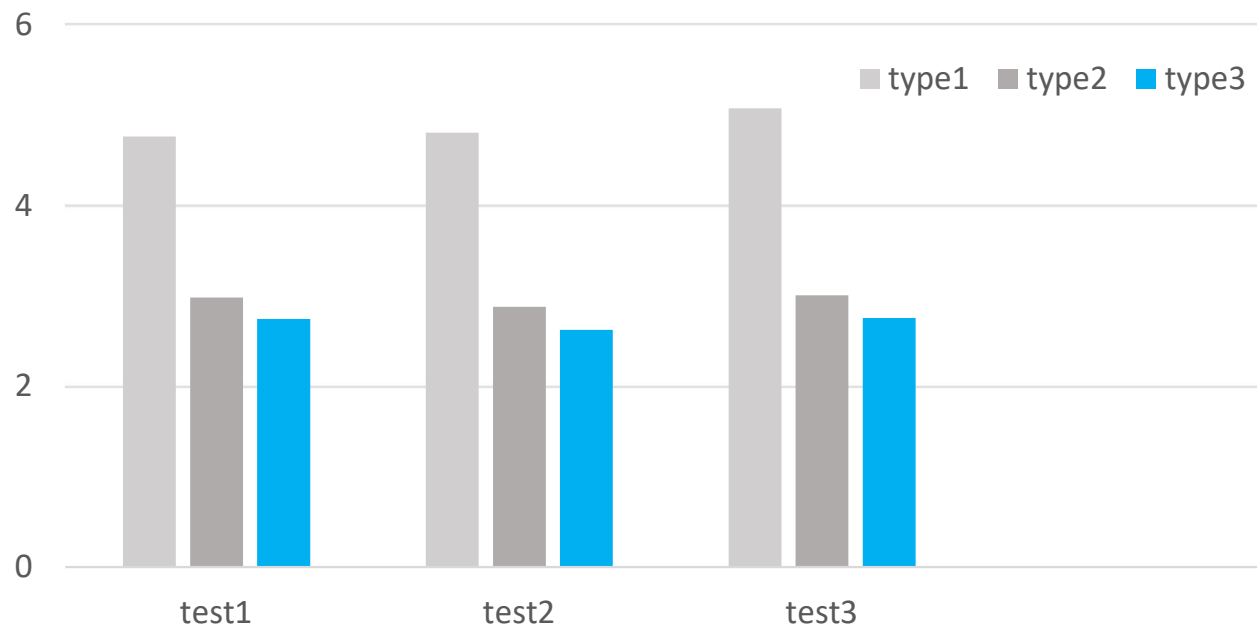
Add Child
[P] TinyIoT
[C] acp_test6...OK
프로그램 수행 시간 :0.000436
```

너무 적은 데이터로 테스트 → 유의미한 차이 없음

▶ ACP_Store 저장 방식 별 실행 시간 비교 (데이터 25,000개)

| | Type1 | Type2 | Type3 |
|-------|----------|----------|----------|
| test1 | 4.758476 | 2.980517 | 2.741837 |
| test2 | 4.803213 | 2.876004 | 2.622257 |
| test3 | 5.071887 | 3.003679 | 2.751947 |

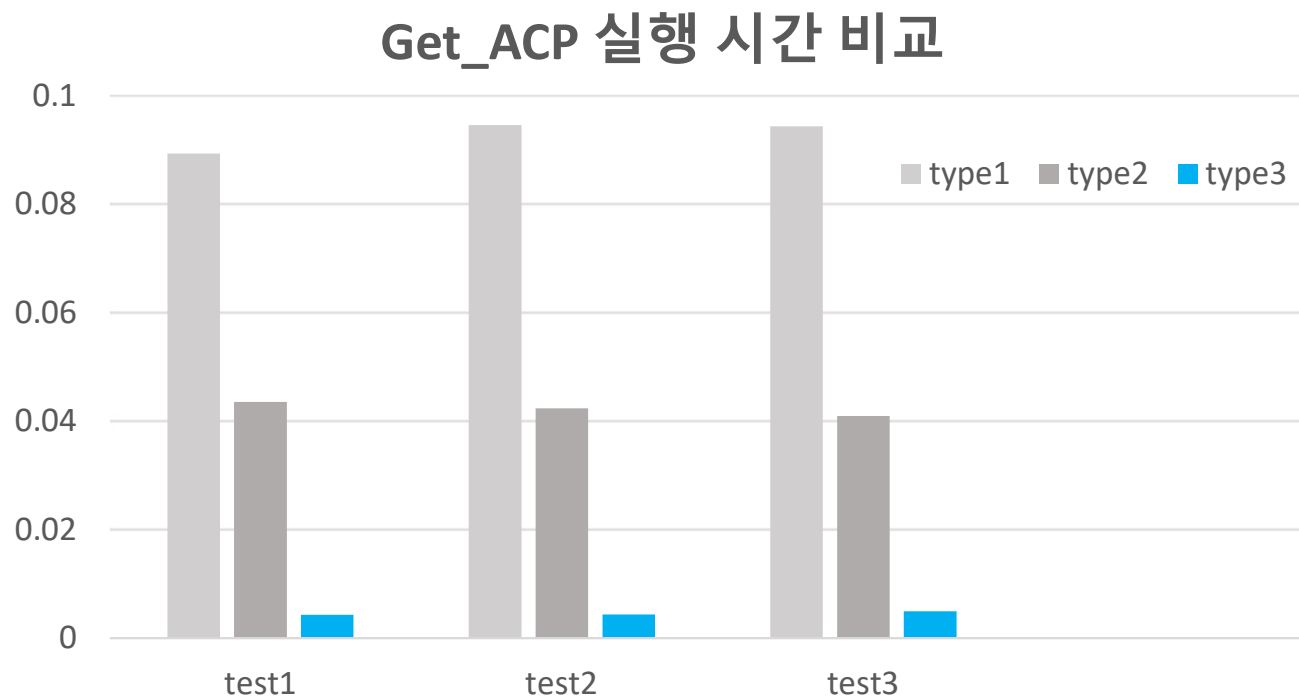
ACP_Store 실행 시간 비교



Type3가 Type1보다 약 1.7배 빠름

▶ Get_ACP 저장 방식 별 실행 시간 비교 (데이터 25,000개)

| | Type1 | Type2 | Type3 |
|-------|----------|----------|----------|
| test1 | 0.089252 | 0.043551 | 0.004315 |
| test2 | 0.094506 | 0.042356 | 0.004391 |
| test3 | 0.094298 | 0.040925 | 0.004939 |



Type3가 Type1보다 약 23배 빠름

▶ ACP CRUD 함수 진행 상황

| Store | Get | Update | Delete |
|--|--|---|--|
| <code>int Store_ACP(ACP *acp_object);</code> | <code>ACP* Get_ACP(char* ri);</code> | <code>int Update_ACP(ACP *acp_object);</code> | <code>int Delete_ACP(char *ri);</code> |
| Type1 | Type1 | Type1 | Type1 |
| Type2 | Type2 | | |
| Type3 | Type3 | | |

Berkeley DB

다음 주 수정할 부분

- I CSE, AE, CNT, CIN 함수 바뀐 저장 방식에 맞게 수정
- II CNT 구조체에 char *acpi 추가했으니 CNT_Store, Get, Update, Delete 함수 수정
- III Get_All_ACP 함수 구현

```
typedef struct {  
    char *et;  
    char *ct;  
    char *lt;  
    char *rn;  
    char *ri;  
    char *pi;  
    char *lbl;  
    char *acpi;  
    int ty;  
    int st;  
    int cni;  
    int cbs;  
} CNT;
```

Thank you