

연구 보고서

작성자	김한호	작성일자	2021.08.22.
1. 연구 계획			
<ul style="list-style-type: none">- 블록생성 시기 측정- 데이터베이스에 인덱스 적용			
2. 논문 연구 진행			
<pre>65 // -Generate all assets 66 const assets = []; 67 for (let i=0; i<testAssetNum; i++) { 68 const asset = {}; 69 asset.docType = baseAsset.docType; 70 asset.byteSize = baseAsset.byteSize; 71 asset.creator = baseAsset.creator; 72 asset.create_at = new Date().toLocaleString(); 73 asset.content = baseAsset.content; 74 asset.uuid = uuidBase + i; 75 assets.push(asset); 76 this.assets.push(asset); 77 }</pre>			
블록생성 시간을 측정하기 위해 성능측정 코드에 생성시간을 저장하는 속성 (asset.create_at)을 추가하여 저장함.			
<pre>112 async submitTransaction() { 113 const uuid = 'client' + this.workerIndex + '_' + this.byteSize + '_' + this.txIndex; 114 //const paddingSize = this.byteSize - 80; 115 this.asset.content = new Date().toLocaleString(); 116 this.txIndex++; 117 const args = { 118 contractId: this.chaincodeID, 119 contractFunction: 'updateAsset', 120 contractArguments: [uuid, this.asset.content], 121 readOnly: false 122 };</pre>			
블록이 업데이트 하는 시간을 저장하는 속성을 기존에 사용하던 필드 (this.asset.content)에 저장하여 업데이트하는 시간을 저장함.			
<pre>2021.08.18-14:50:10.055 info [caliper] [caliper-worker] Info: worker 0 prepare test phase for round 4 is starting... -> Creating asset set of byte size: 100</pre>			
성능측정 시간을 하는 로그에서 2021.08.18-14:50:10.055라는 시간에 블록을 생성하는 시간을 확인할 수 있음.			
<pre>2021.08.18-14:50:12.194 info [caliper] [caliper-worker] Info: worker 0 prepare test phase for round 4 is completed</pre>			
성능측정 코드가 완료되는 순간이 2021.08.18-14:50:12.194로 확인됨.			

<pre>{ "id": "client0_100_0", "key": "client0_100_0", "value": { "rev": "113-3727763f22acaaa44e042bf28" }, "doc": { "_id": "client0_100_0", "_rev": "113-3727763f22acaaa44e042bf28", "byteSize": 100, "content": "8/18/2021, 2:50:12 PM", "create_at": "8/18/2021, 2:50:10 PM", "creator": "client0", "docType": "fixed-asset", "uuid": "client0_100_0", "~version": "CgMBrQA=" } }</pre>	<pre>{ "id": "client0_100_99", "key": "client0_100_99", "value": { "rev": "2-1e3af11bac08e5e0a6f8284475de56d6" }, "doc": { "_id": "client0_100_99", "_rev": "2-1e3af11bac08e5e0a6f8284475de56d6", "byteSize": 100, "content": "8/19/2021, 11:04:00 AM", "create_at": "8/19/2021, 11:02:15 AM", "creator": "client0", "docType": "fixed-asset", "uuid": "client0_100_99", "~version": "CgMB1gA=" } }</pre>
---	---

블록생성 시간은 블록이 생성되는 시간이 2021.08.18-14:50:10으로 표시되는 것을 확인할 수 있음. 성능측정 코드에서 시간을 입력했기 때문에 체인코드가 생성되는 순간을 정확하게 알아 낼 수 없다고 판단함.

```
64     asset['create_at'] = new Date().toLocaleString();
65     await ctx.stub.putState(uuid, Buffer.from(JSON.stringify(asset)));
```

체인코드에서 블록을 생성하기 직전에 시간을 저장하여 체인 생성시간을 더 정확하게 판단할 수 있도록 작성함.

```
277     asset[attr] = newValue;
278     asset['update_at'] = new Date().toLocaleString();
279     return await ctx.stub.putState(uid, Buffer.from(JSON.stringify(asset)));
280 }
```

체인코드를 정보를 업데이트 하는 시간 직전에 정보를 저장하여 정확한 체인 생성 시기를 측정할 수 있도록 함.

```
17   for container in containers_list:
18       print(container.id)
19       os.system("docker cp /usr/share/zoneinfo/Asia/Seoul {}:etc/localtime".format(container.id))
20       os.system("docker exec -it {} date".format(container.id))
21
```

도커 컨테이너의 시간이 미국 기준시간을 바꿔 들어가 한국 시간으로 변경하여 시스템 시간과 동기화를 시킴.

```
634ce865b9b6f80013a7615a4560eceba7e96eb7fb8fc700638b65b5f07a7358
Sat Aug 21 10:30:30 UTC 2021
Sat Aug 21 19:30:30 KST 2021
e94495f8c19b08845e46f7a014387384b9a0db9bf549d73a98c7a0a2432b75f3
Sat Aug 21 10:30:30 UTC 2021
Sat Aug 21 19:30:31 KST 2021
```

컨테이너가 미국시간(UTC)에서 한국 표준시간(KST)로 변하는 것을 확인함.

```

- label: update-asset-evaluate-1-100-100-100
  chaincodeID: fixed-asset
  txNumber: 100
  rateControl:
    type: fixed-rate
    opts:
      tps: 1
  workload:
    module: benchmarks/api/fabric/lib/update-asset.js
    arguments:
      chaincodeID: fixed-asset
      create_sizes:
        - 100
      noSetup: false
      assets: 100

```

보내는 속도는 1tps로 하고 100개의 업데이트 트랜잭션을 보내서 생성시간과 업데이트 시간을 확인함.

"ilient0_160_0"	"ilient0_160_99"
<pre> { "id": "ilient0_160_0", "key": "ilient0_160_0", "value": { "rev": "2-d2e82ad50217a2f8d9e36e1f366" }, "doc": { "_id": "ilient0_160_0", "_rev": "2-d2e82ad50217a2f8d9e36e1f36", "content": "a", "create_at": "8/22/2021, 8:25:32 PM", "ibyteSize": 160, "icreator": "ilient0", "idocType": "index-asset", "uid": "ilient0_160_0", "update_at": "8/22/2021, 8:25:37 PM", "~version": "CgMBCwA=" } } </pre>	<pre> { "id": "ilient0_160_99", "key": "ilient0_160_99", "value": { "rev": "2-b62814d437f88014cd1cd7c4427" }, "doc": { "_id": "ilient0_160_99", "_rev": "2-b62814d437f88014cd1cd7c442", "content": "a", "create_at": "8/22/2021, 8:25:35 PM", "ibyteSize": 160, "icreator": "ilient0", "idocType": "index-asset", "uid": "ilient0_160_99", "update_at": "8/22/2021, 8:27:16 PM", "~version": "CgMBMgA=" } } </pre>

```

2021.08.22-20:25:32.430 info [caliper] [connectors/v2/FabricGateway] Generating contract map for user _Org2MSP_user1.org2.example.com

```

100개의 Asset을 생성하고 처음 생성된 Asset은 처음 성능 측정 코드가 실행될 때 생성되는 것을 확인할 수 있음.

```
2021.08.22-20:25:32.474 info [caliper] [caliper-worker] Info: worker 0 p
repare test phase for round 0 is starting...
-> Creating asset set of byte size: 160
2021.08.22-20:25:37.223 info [caliper] [caliper-worker] Info: worker 0 p
repare test phase for round 0 is completed
```

생성을 끝내고 업데이트 트랜잭션을 하기 직전에 마지막 Asset이 생성된 것으로 판단됨.

```
2021.08.22-20:25:37.229 info [caliper] [caliper-worker] Worker #0 starti
ng workload loop
```

성능측정이 시작되는 시간과 첫 Asset 업데이트 시간이 동일하다는 것을 확인할 수 있음.

```
2021.08.22-20:27:12.265 info [caliper] [default-observer] [update-asset-ev
aluate-1-100-100-100 Round 0 Transaction Info] - Submitted: 95 Succ: 94 Fail:0 U
nfinished:1
```

```
-11
-11
-11
-11
```

```
2021.08.22-20:27:17.268 info [caliper] [default-observer] [update-asset-ev
aluate-1-100-100-100 Round 0 Transaction Info] - Submitted: 100 Succ: 99 Fail:0
Unfinished:1
```

트랜잭션 성능 측정이 끝나기 직전에 마지막 Asset의 업데이트 시간이 변경되었다는 것을 예측할 수 있음.

결론: 블록 생성시기는 블록이 생성되는 함수가 작동하면 생성하는 것을 확인할 수 있음. 블록 업데이트 시간 또한 블록이 업데이트하는 트랜잭션 발생하면 업데이트를 함.

```
{"index":{"fields":["uid"],"ddoc":"indexOwner1Doc", "name":"indexOwner1","type":"json"},
{"index":{"fields":["uid","idocType"],"ddoc":"indexOwner2Doc", "name":"indexOwner2","type":
{"index":{"fields":["idocType","icreator","ibyteSize"],"ddoc":"indexOwner3Doc", "name":"inde
```

couchdb를 실행할 때 /META-INF/디렉토리 밑에 인덱스를 저장하고 싶은 코드를 작성하여 위치시켜서 실행함. 인덱스가 입력되었다는 로그를 확인하지 못함. 다른 방식을 이용하여 작성하기로 함.

```

{} indexUid.json ×
META-INF > statedb > couchdb > indexes > {} indexUid.json > ...
1  {
2    "index":{
3      "fields":["uid"]
4    },
5    "ddoc":"index1Doc",
6    "name":"index1",
7    "type":"json"
8  }

{} indexUidType.json ×
META-INF > statedb > couchdb > indexes > {} indexUidType.json > name
1  {
2    "index":{
3      "fields":["uid","idocType"]
4    },
5    "ddoc":"index2Doc",
6    "name":"index2",
7    "type":"json"
8  }

{} indexTypeCreSize.json ×
META-INF > statedb > couchdb > indexes > {} indexTypeCreSize.json > type
1  {
2    "index":{
3      "fields":["idocType","icreator","ibyteSize"]
4    },
5    "ddoc":"index3Doc",
6    "name":"index3",
7    "type":"json"
8  }

```




인덱스를 저장하고 싶은 필드와 구별자(ddoc)와 이름(name) 파일형식(json)을 입력하여 실행을 시킴.

```

2021-08-20 06:52:15.219 UTC [couchdb] createIndex -> INFO 071 Created CouchDB index [index3] in state
database [mychannel_index-asset] using design document [_design/index3Doc]
2021-08-20 06:52:15.219 UTC [statecouchdb] ProcessIndexesForChaincodeDeploy -> INFO 072 successfully
submitted index creation request present in the file [META-INF/statedb/couchdb/indexes/indexTypeCreSi
ze.json] for chaincode [index-asset] on channel [mychannel]
2021-08-20 06:52:15.270 UTC [couchdb] createIndex -> INFO 073 Created CouchDB index [index1] in state
database [mychannel_index-asset] using design document [_design/index1Doc]
2021-08-20 06:52:15.270 UTC [statecouchdb] ProcessIndexesForChaincodeDeploy -> INFO 074 successfully
submitted index creation request present in the file [META-INF/statedb/couchdb/indexes/indexUid.json]
for chaincode [index-asset] on channel [mychannel]
2021-08-20 06:52:15.337 UTC [couchdb] createIndex -> INFO 075 Created CouchDB index [index2] in state
database [mychannel_index-asset] using design document [_design/index2Doc]
2021-08-20 06:52:15.337 UTC [statecouchdb] ProcessIndexesForChaincodeDeploy -> INFO 076 successfully
submitted index creation request present in the file [META-INF/statedb/couchdb/indexes/indexUidType.j
son] for chaincode [index-asset] on channel [mychannel]

```

피어 노드에서 couchdb를 확인하여 인덱스가 입력된 것을 확인할 수 있었음.

Document ID		Options	{ } JSON		
<input type="checkbox"/>	Table	Metadata	{ } JSON		Create Document
id	key	value			
<input type="checkbox"/>  _design/index1Doc	_design/index1Doc	{ "rev": "1-f8e151e49a73322afc...			
<input type="checkbox"/>  _design/index2Doc	_design/index2Doc	{ "rev": "1-aa3fa360239942ac48...			
<input type="checkbox"/>  _design/index3Doc	_design/index3Doc	{ "rev": "1-537a6d77fe29acf66dd...			

입력된 인덱스 내용은 데이터베이스에 확인할 수 있음.

```
normal-asset$ ./update-public-normal-asset-below-1000.sh >
result_fixed.txt & ../../../../index-asset/update/update-public-normal-asset/update-
-public-normal-asset-below-1000.sh > ./result_index.txt && fg
```

위의 명령어를 통해서 인덱스를 적용한 시스템과 적용하지 않은 시스템의 테스트를 진행함.

```
date-asset-evaluate-1-100-100-100) in 69.982 seconds
date-asset-evaluate-1-100-100-100) in 69.976 seconds
```

적용하지 않은 시스템과 적용한 시스템이 거의 동일한 측정값을 보인 것을 확인할 수 있었음.

```
h2test@h2test-VirtualBox:~$ curl -X GET http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/
{"total_rows":2,"indexes":[{"ddoc":null,"name":"_all_docs","type":"special","def":{"fields":[{"_id":"on","partitioned":false,"def":{"fields":[{"foo":"asc"}]}}]}
```

필드 구별자를 선택할 수 없는 _all_docs라는 인덱스가 _id라는 필드를 오름차순으로 인덱스 하는 것을 확인할 수 있었음.

```
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_0/json/_all_docs' -H
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_json/_all_docs' -H
"reason":"function_clause","ref":487253626}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_\x0/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_\x00/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_\0/json/_all_docs' -H
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_\000/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_\x00/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_Z/json/_all_docs' -H
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_u0000/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_%00/json/_all_docs'
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_^@/json/_all_docs' -H
son":"missing"}
~$ curl -X DELETE 'http://admin:adminpw@localhost:5984/mychannel_fixed-asset/_index/_U+0000/json/_all_docs'
son":"missing"}
~$
```

null문자를 입력하는 모든 시도를 했지만 ddoc가 없는 것을 접근할 수 없었음.
자동으로 정렬되는 인덱스가 적용되어 있다면 순서대로 값이 입력되지 않는다면

인덱스를 적용하지 않는 것과 같지 않겠냐는 생각이 듭.

```
module.exports.retrieveRandomAssetIds = function(assetNumber) {
  const uuids = [...Array(assetNumber).keys()];
  // shuffle array using Fisher-Yates shuffle
  for (let i = uuids.length - 1; i > 0; i--) {
    let j = Math.floor(Math.random() * (i + 1));
    // swap elements uuids[i] and uuids[j]
    [uuids[i], uuids[j]] = [uuids[j], uuids[i]];
  }
  return uuids;
}
```

asset의 개수를 입력하면 랜덤으로 숫자가 나오는 함수를 asset에 적용하기로 함.

```
const assets = [];
this.assets = helper.retrieveRandomAssetIds(testAssetNum);
```

인덱스를 순차적으로 저장하지 않도록 미리 식별자를 랜덤으로 가져와서 저장함. 인덱스가 적용되는 순차적으로 들어가는 성능측정 코드와 적용되지 않는 성능 측정 코드를 작성하고 피어에 따로 동시에 수행함.

```
ate-asset-evaluate-1-100-100-100) in 6.009 seconds
```

```
date-asset-evaluate-1-100-100-100) in 5.827 seconds
```

100개의 트랜잭션을 수행했을 때는 별 차이가 없는 것으로 확인됨.

```
2021.08.22-20:06:26.983 [32m info ^[[39m [caliper] [round-orchestrator] Finished round 1 (up
date-asset-evaluate-1-100-100-100) in 34.41 seconds
2021.08.22-20:06:26.985 [32m info ^[[39m [caliper] [monitor.js] Stopping all monitors
2021.08.22-20:06:26.986 [32m info ^[[39m [caliper] [report-builder] ### All test results ###
2021.08.22-20:06:26.993 [32m info ^[[39m [caliper] [report-builder]
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
sult_fixed.txt
```

```
310
317 2021.08.22-20:05:39.152 [32m info ^[[39m [caliper] [round-orchestrator] Finished round 1 (up
ate-asset-evaluate-1-100-100-100) in 12.415 seconds
318 2021.08.22-20:05:39.155 [32m info ^[[39m [caliper] [monitor.js] Stopping all monitors
319 2021.08.22-20:05:39.155 [32m info ^[[39m [caliper] [report-builder] ### All test results ###
320 2021.08.22-20:05:39.158 [32m info ^[[39m [caliper] [report-builder]
321 +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
result_index.txt
```

1000개의 트랜잭션을 수행했을 때는 랜덤으로 들어가 인덱스가 적용되지 않는 체인코드가 느리게 실행되는 것으로 확인함. 임의적으로 시스템 인덱스를 무시하게 하는 실험이었기 때문에 이 방향이 맞는지 확인해서 진행해야 할 것으로 생각됨.

3. 다음 계획

- 생성한 체인코드와 동일한 실험 가정을 통해 가지고 실험 값에 따른 결과 데이터 생성 진행.