공공 API 프로젝트

## 개요

1. **프로젝트 목적**

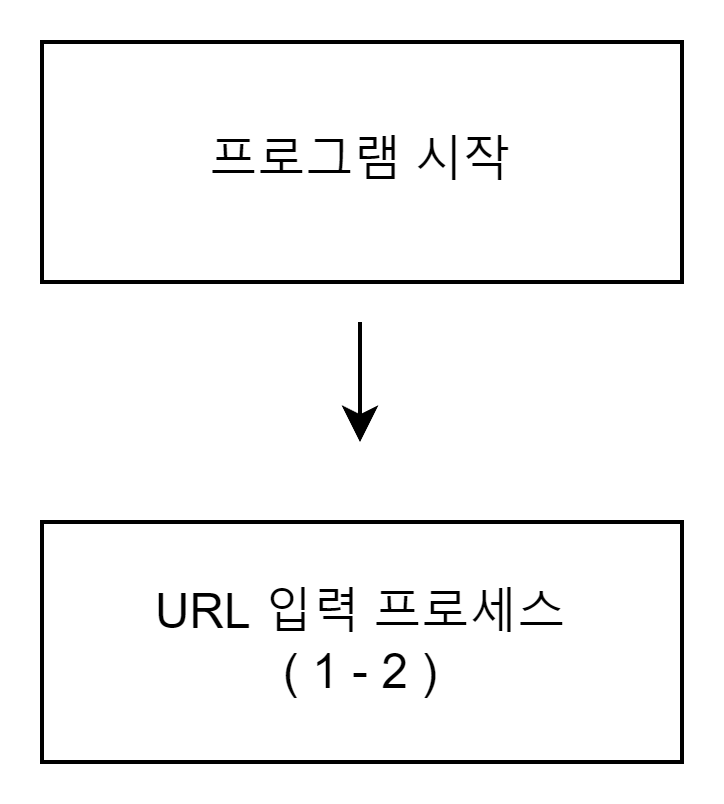
공공 API를 호출하여 응답데이터(Body)를 DB에 저장

1. **사용자 입력 데이터 종류**
2. 요청 URL
3. Header 정보
4. Parameter정보
5. Body 정보
6. Method
7. 확장자
8. 컬럼명
9. 컬럼 데이터 타입 및 크기
10. 제약 사항
11. 서비스 유형을 REST API로 제한 ( Method : GET / POST )
12. 확장자를 XML과 JSON으로 제한
13. 입출력 데이터의 구조에서 서브는 지원하지 않는다.
14. 입력 바디의 list는 지원하지만, body 테이블 value값 입력시 [“val1”, “val2”, …]형식으로 입력 해야 한다.
15. 구현하지 않은 프로세스
16. 수정 프로세스
17. 등록 프로세스
18. Test 주소

## 프로세스

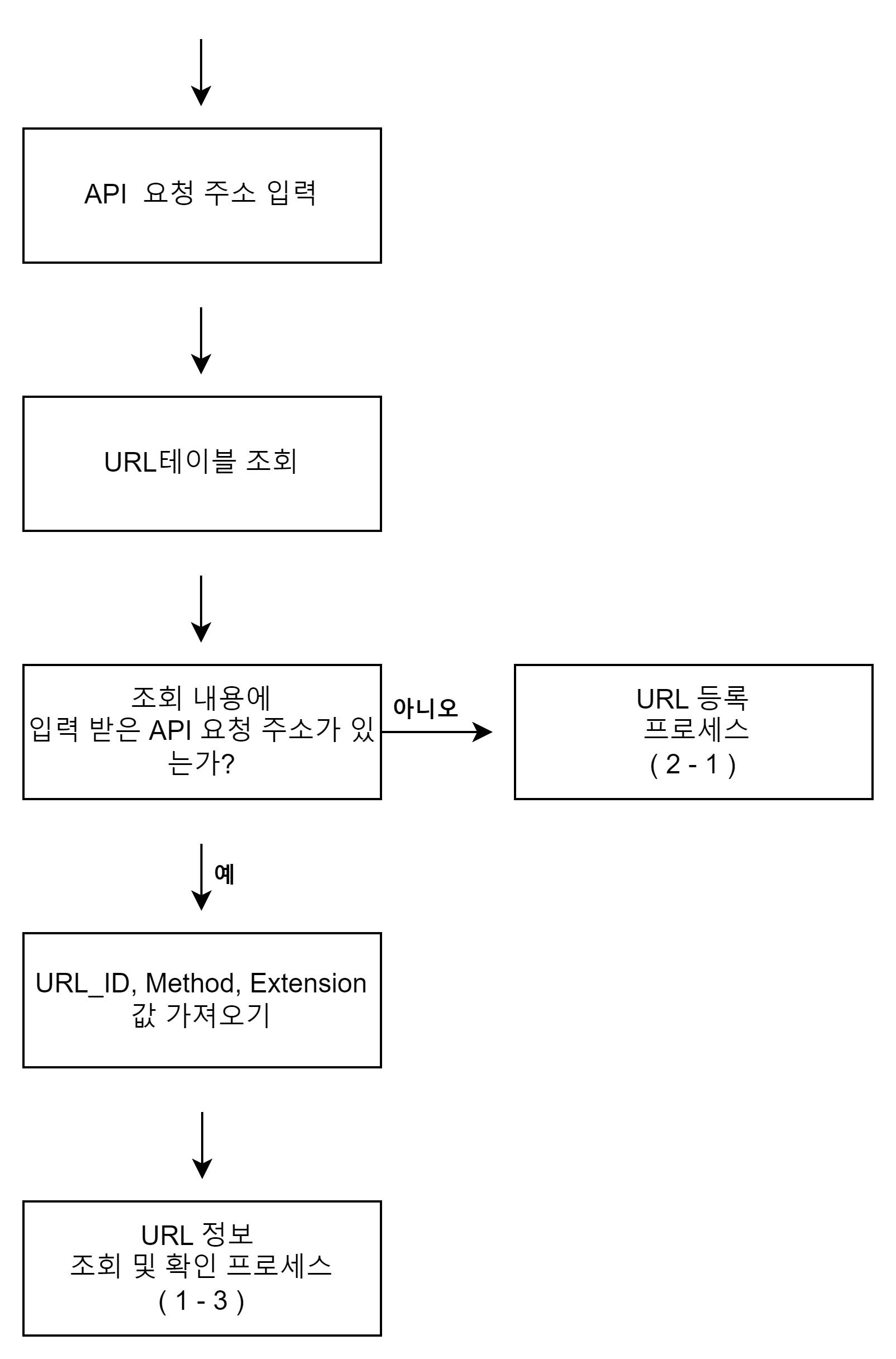
### 정보 확인 프로세스

#### 프로그램 시작

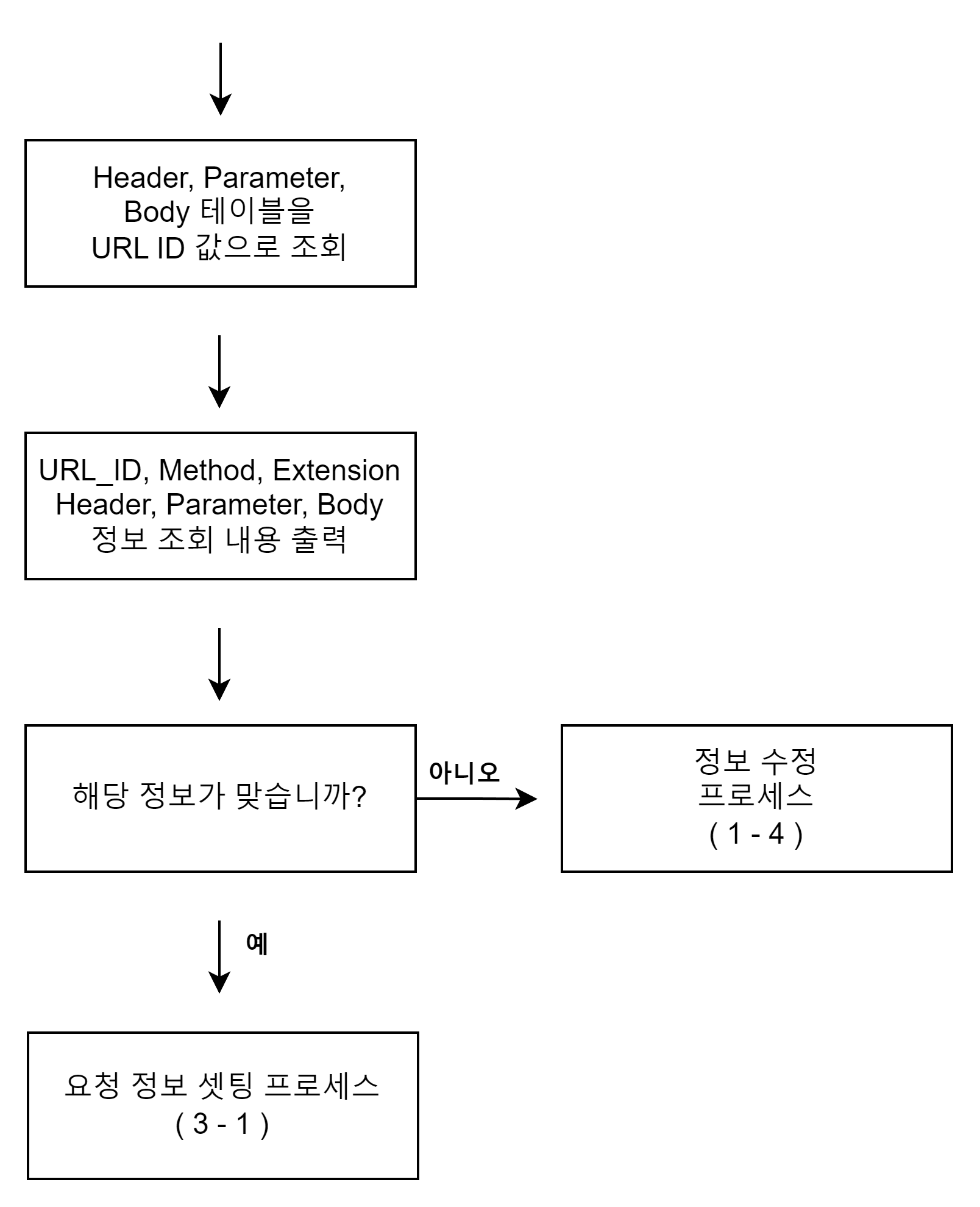


%

#### URL 입력 프로세스



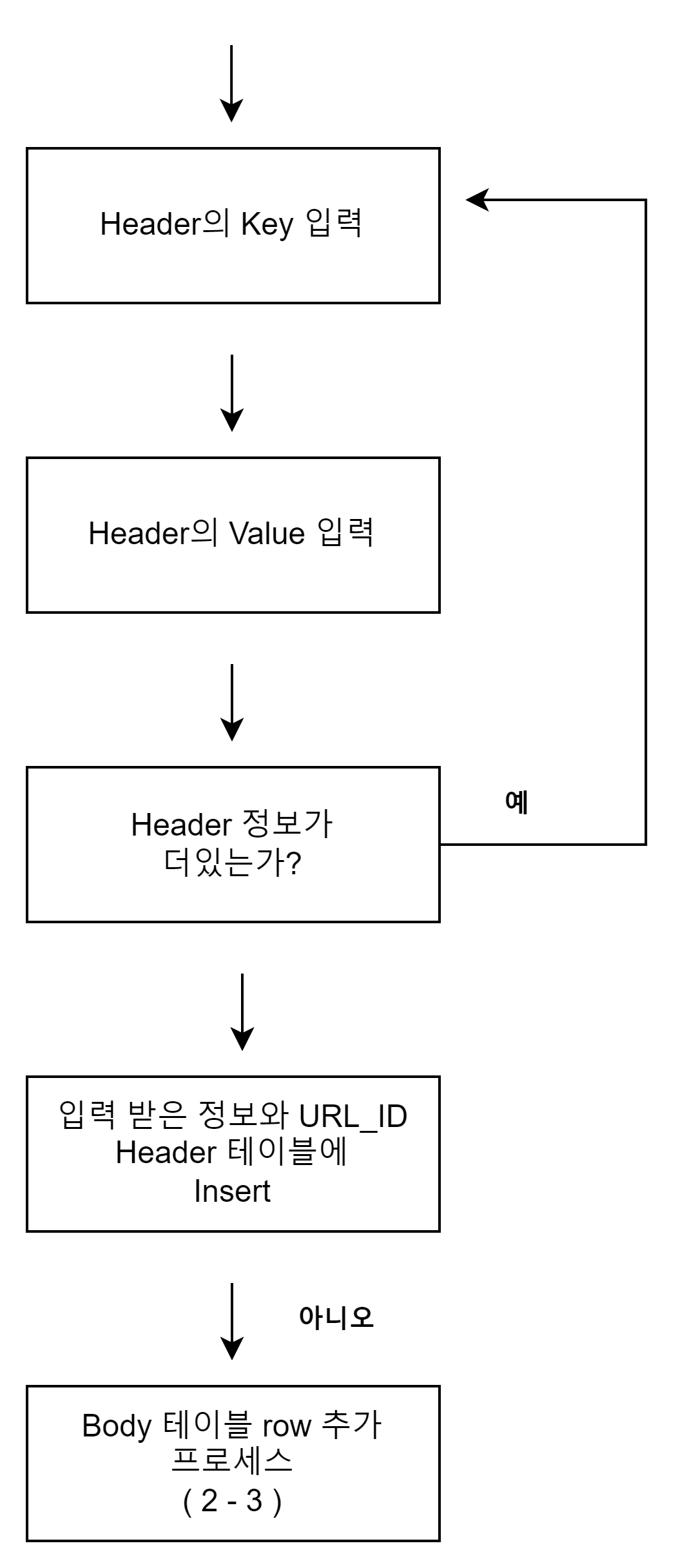
#### URL 정보 조회 및 확인 프로세스



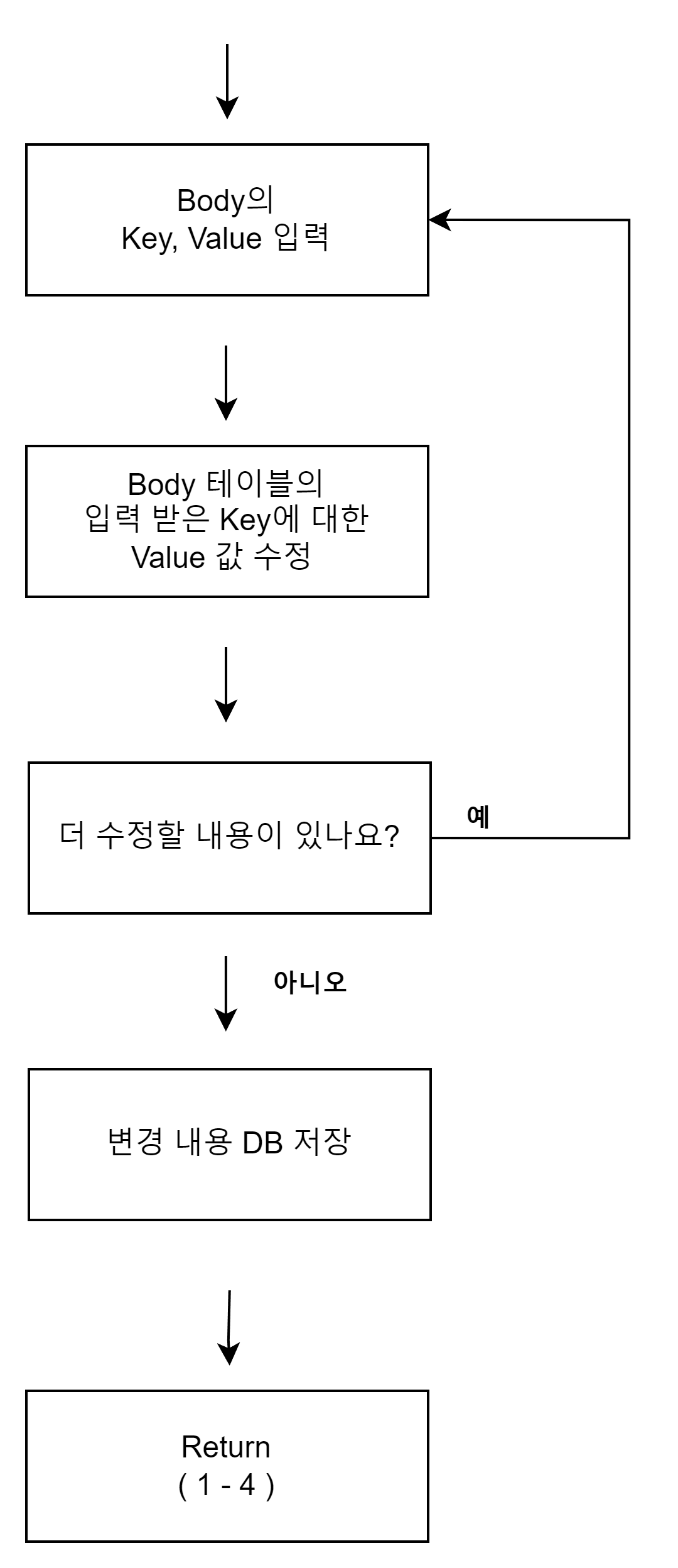
#### 정보 수정 프로세스



#### Header 정보 수정 프로세스



#### Body 정보 수정 프로세스

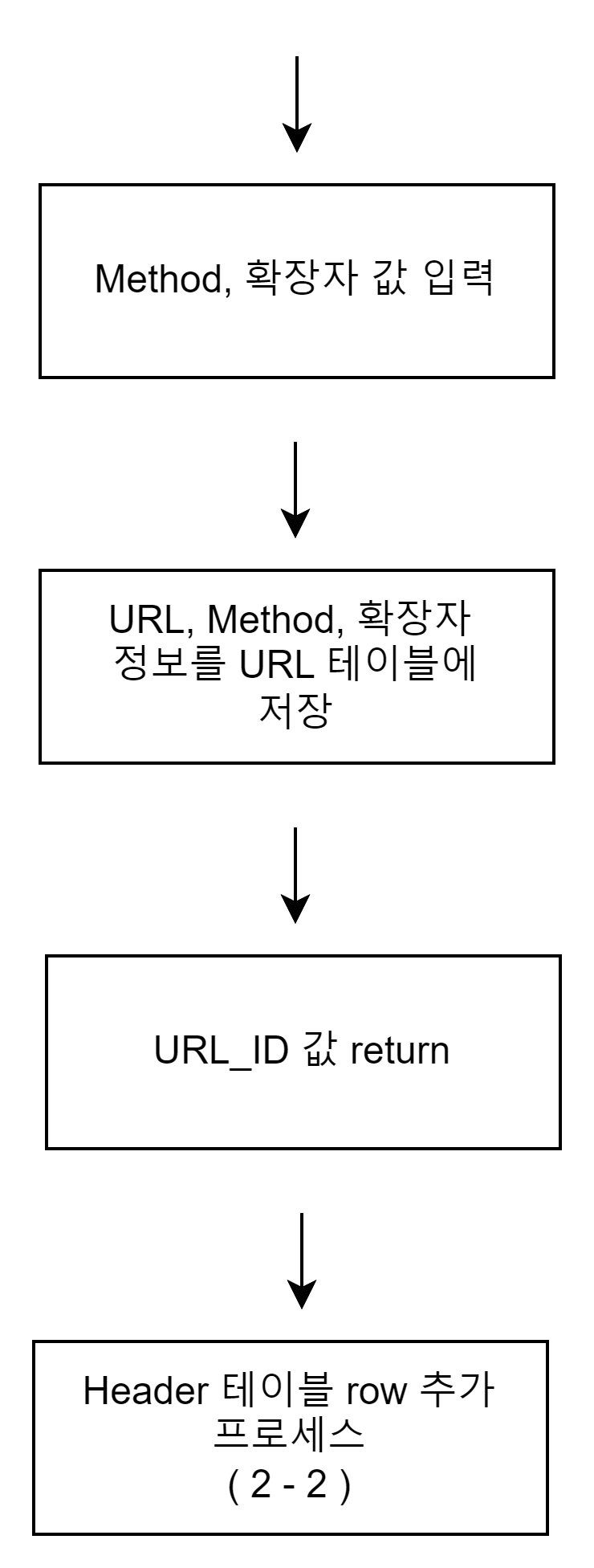


#### 메소드 정보 확인 및 수정 프로세스

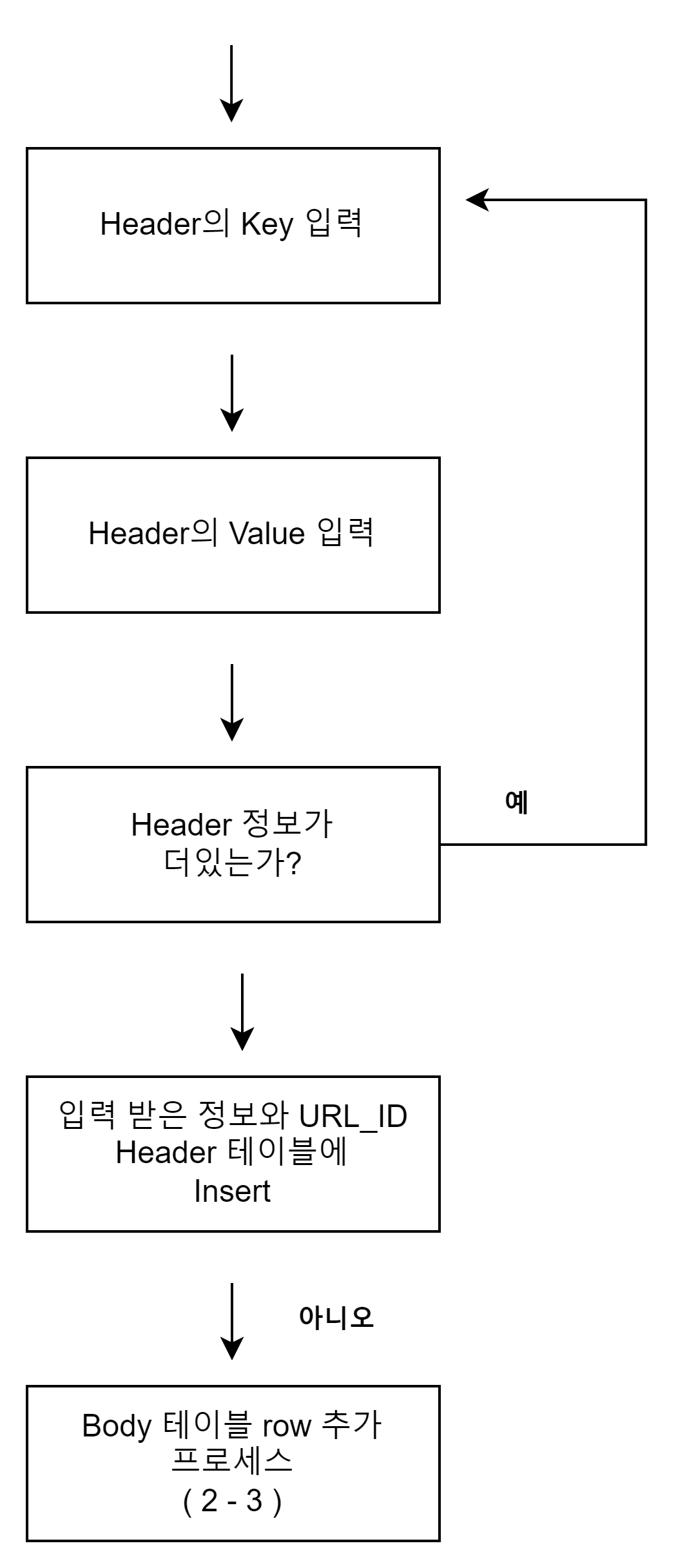


### URL등록 및 테이블 생성 프로세스

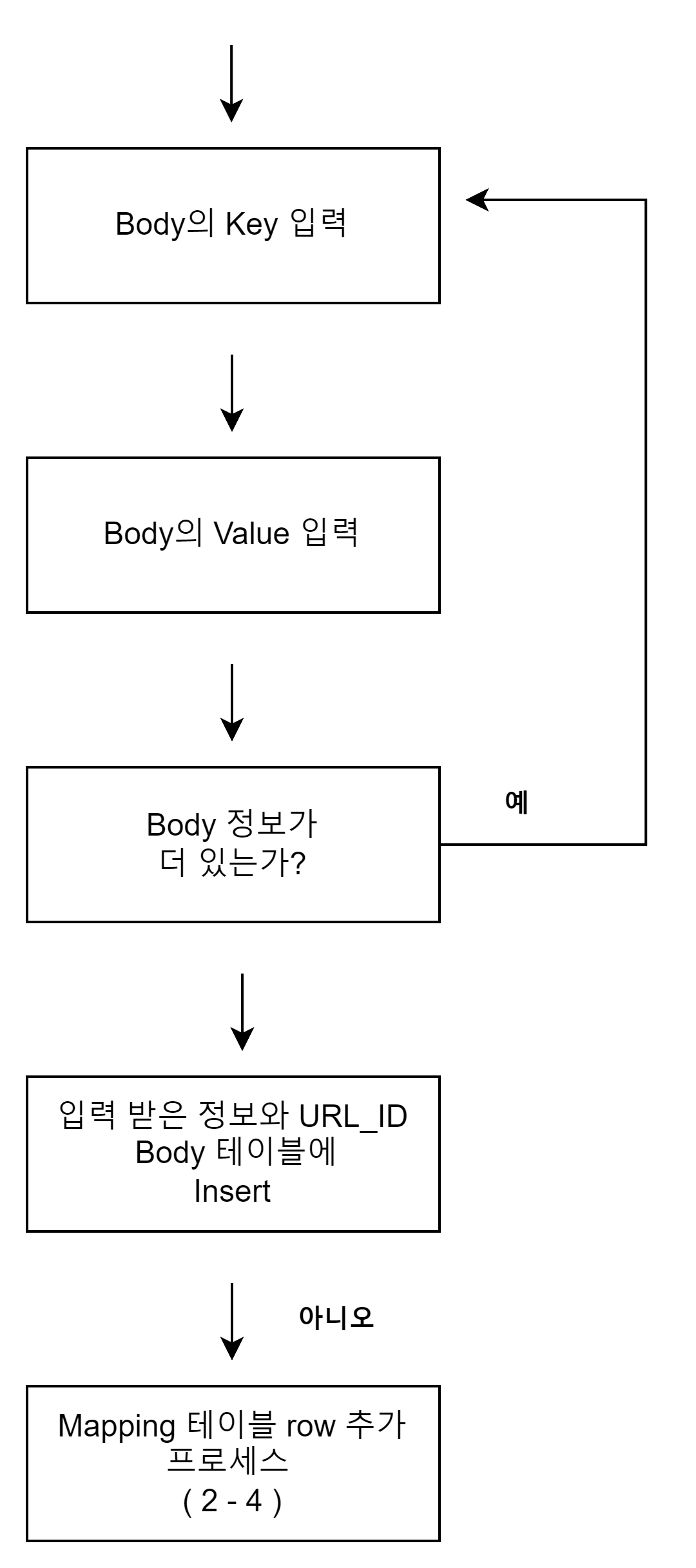
#### URL 등록 프로세스



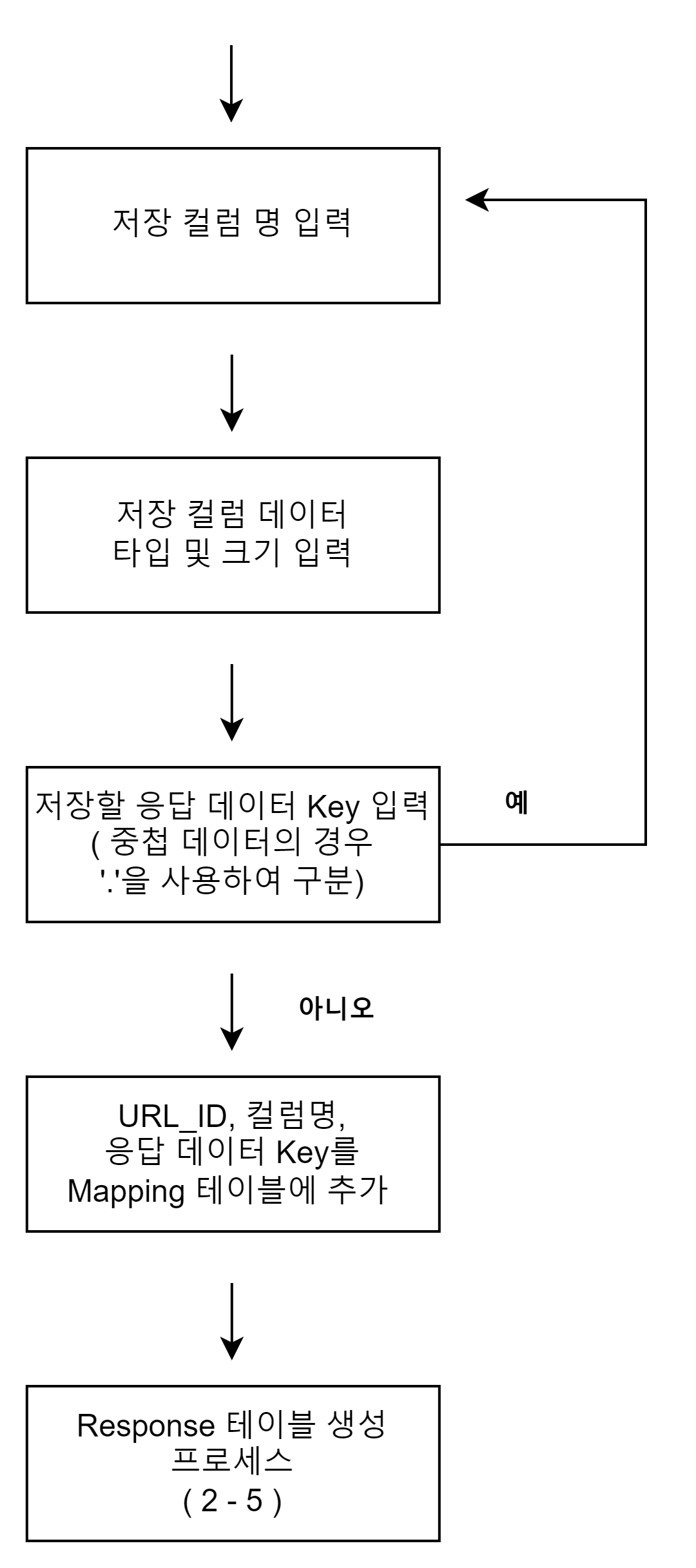
#### Header 테이블 row 추가 프로세스



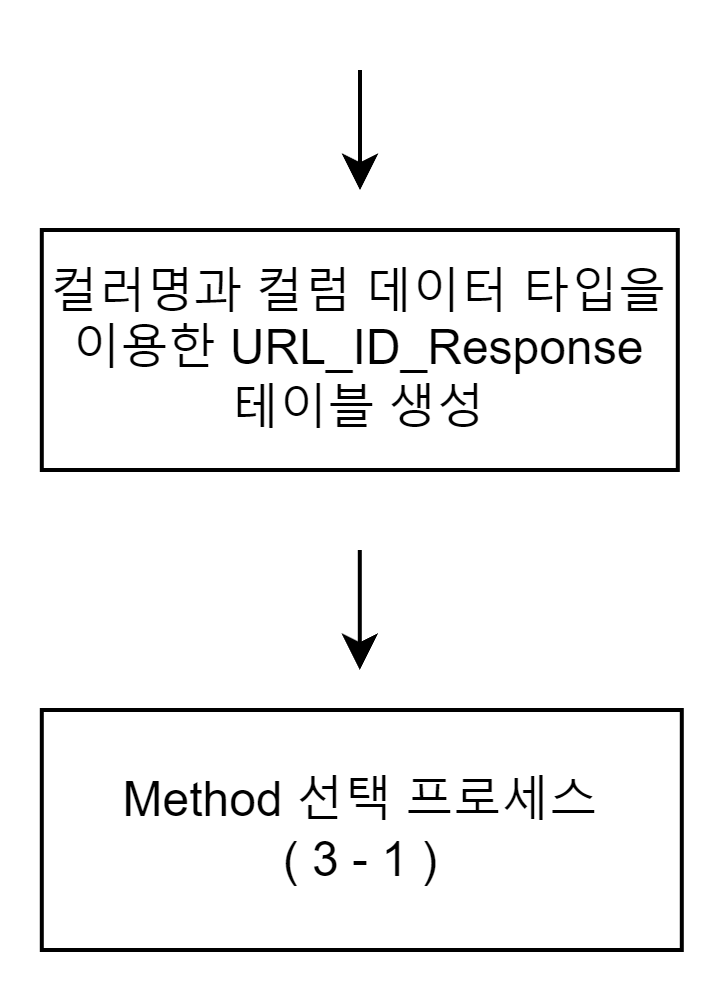
#### Body 테이블 row 추가 프로세스



#### Mapping 테이블 추가 프로세스

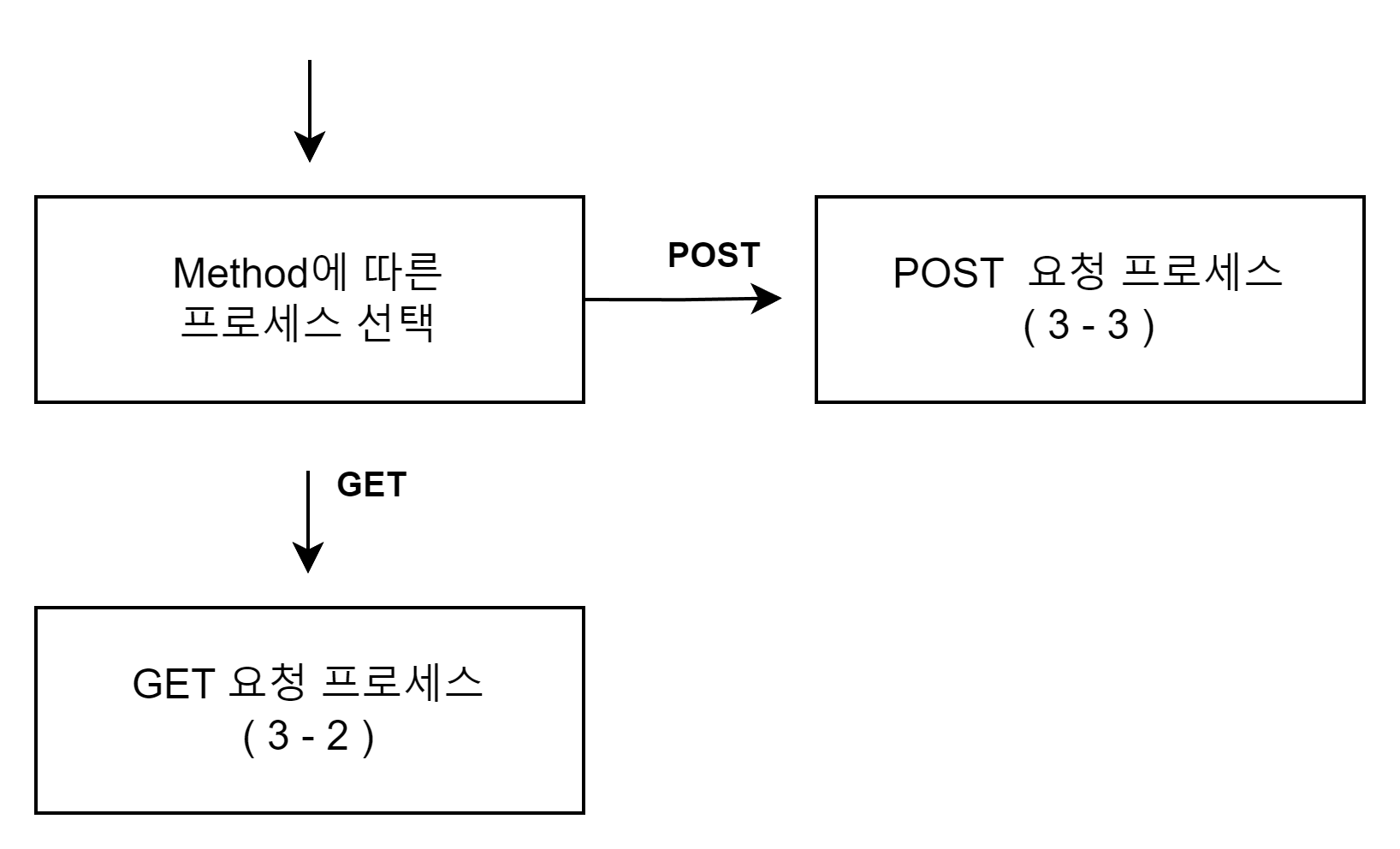


#### Response 테이블 생성 프로세스

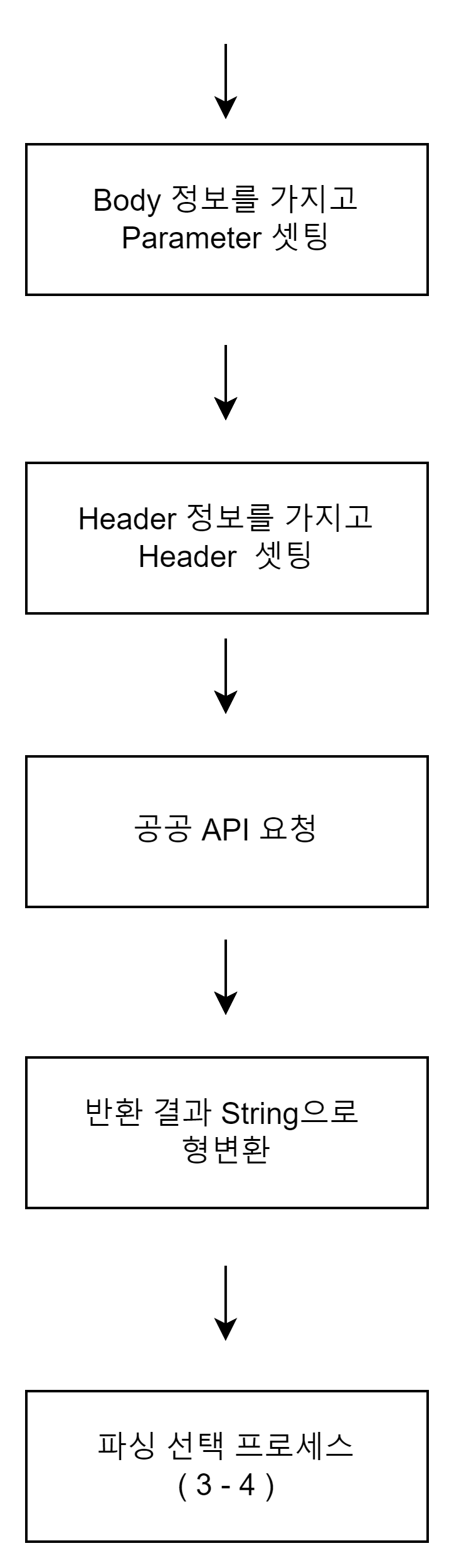


### 공공 API 요청 및 파싱 프로세스

#### Method 선택 프로세스



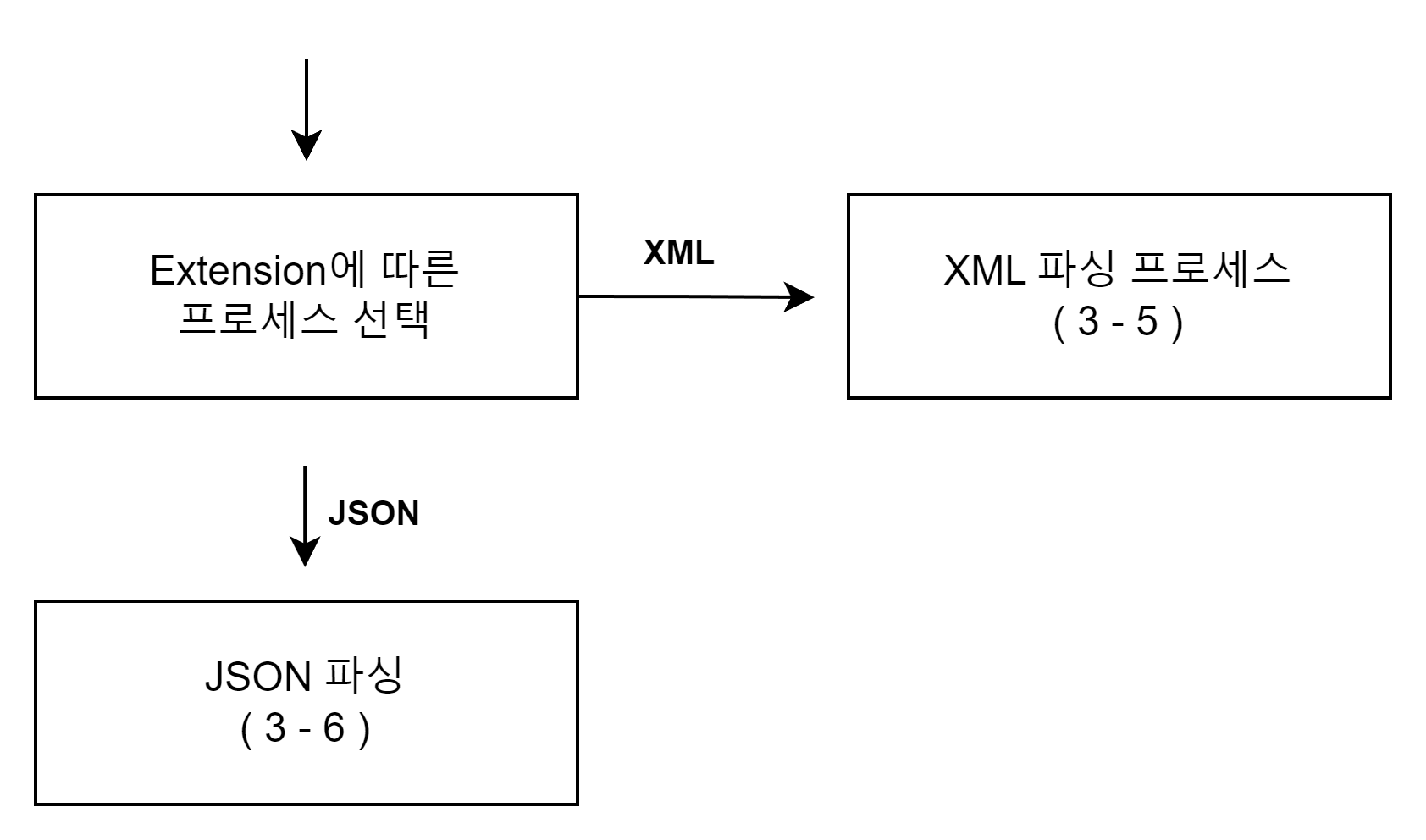
#### GET 요청 프로세스



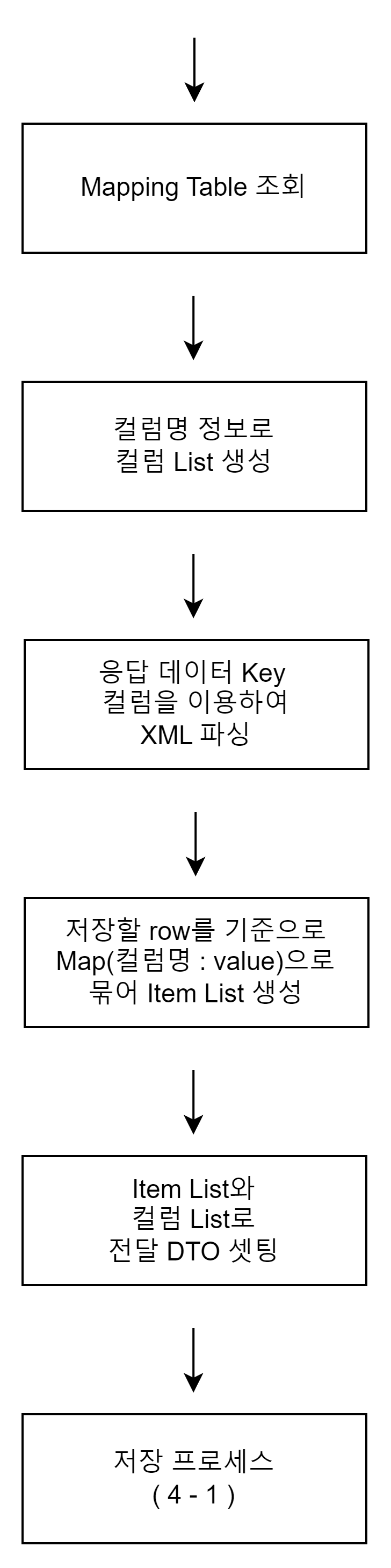
#### POST 요청 프로세스



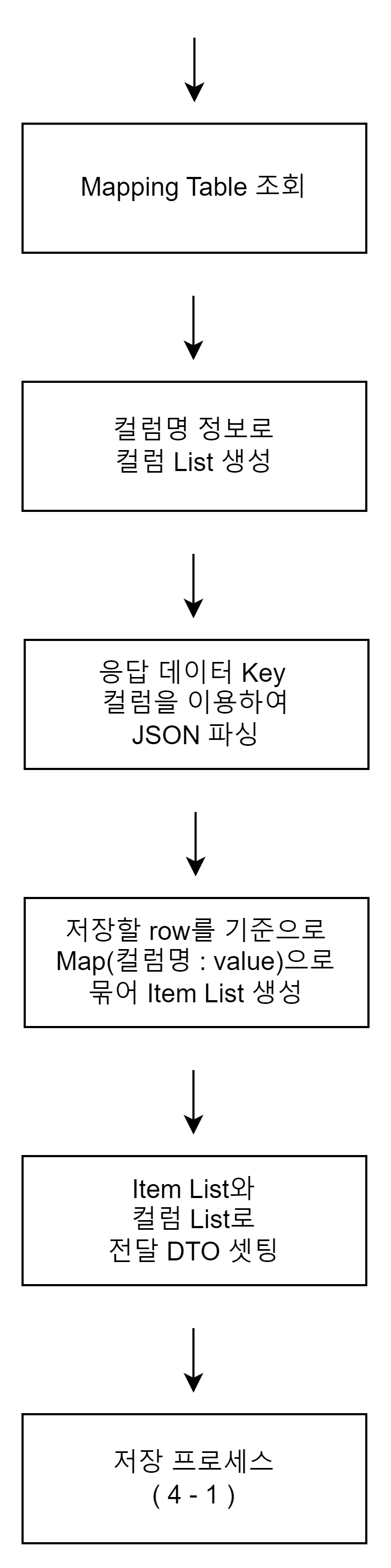
#### 파싱 선택 프로세스



#### XML 파싱 프로세스

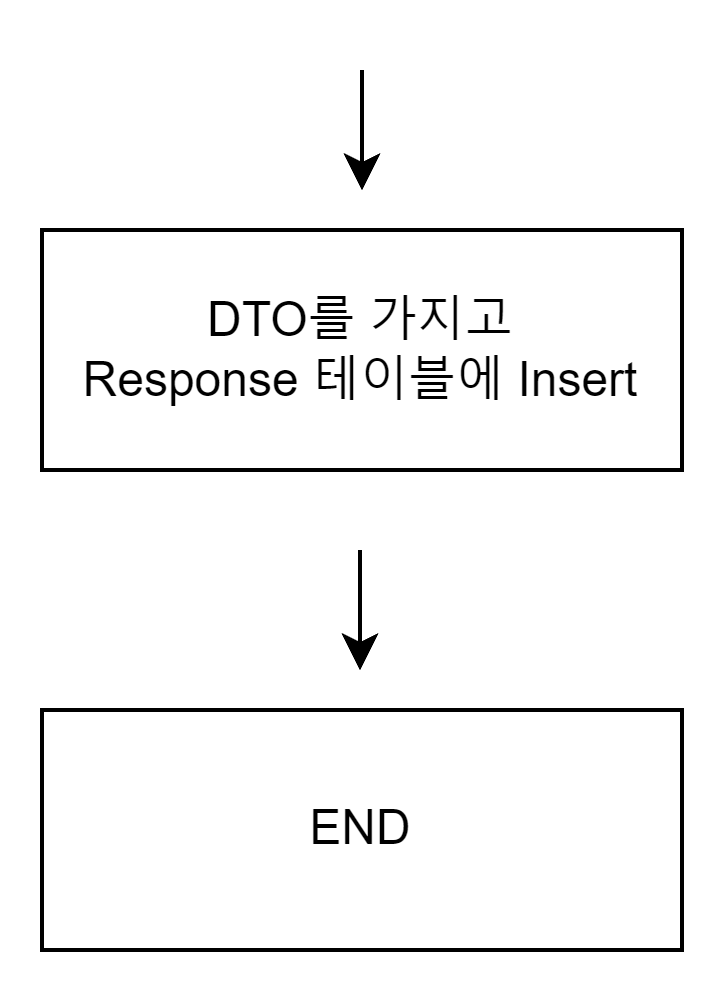


#### JOSN 파싱 프로세스



### 저장 프로세스

#### 저장 프로세스



## DB 스키마

### URL 테이블 스키마

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **테이블 명** | | URL | | |
| **COL 이름** | **데이터 타입** | | **PK 여부** | **NOT NULL 여부** |
| urlId | serial | | Y | Y |
| url | varchar(2048) | |  | Y |
| method | int( 0또는 1) | |  | Y |
| extension | int( 0또는 1) | |  | Y |

1. urlId: url 별로 고유한 테이블 이름을 갖도록 Response 테이블에 이용한다.
2. url : 최대길이와 유사한 길이로 최대 길이를 2048으로 설정
3. Method : 0 -> GET, 1 -> POST
4. Extension : 0 -> xml, 1 -> JSON

### Header 테이블 스키마

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **테이블 명** | | Header | | |
| **COL 이름** | **데이터 타입** | | **PK 여부** | **NOT NULL 여부** |
| urlId | int | | Y | Y |
| key | varchar(1024) | | Y | Y |
| value | varchar(1024) | |  | Y |

1. urlId와 key를 PK로 사용
2. key와 value의 길이는 1024로 제한

### Body테이블 스키마

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **테이블 명** | | Parameter | | |
| **COL 이름** | **데이터 타입** | | **PK 여부** | **NOT NULL 여부** |
| urlId | int | | Y | Y |
| key | varchar(256) | | Y | Y |
| value | varchar(4000) | |  | Y |
| isList | int | |  | Y |

1. urlId와 key를 PK로 사용
2. key 256로 제한
3. value 4000로 제한
4. isList : 1 -> lsit, 0-> list(x)
5. value값이 list인 경우 [“val1”, “val2”, “val3”, …]으로 입력

### Parent 테이블 스키마

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **테이블 명** | | Parent | | |
| **COL 이름** | **데이터 타입** | | **PK 여부** | **NOT NULL 여부** |
| urlId | int | | Y | Y |
| parent | varchar(1024) | |  |  |

1. Parent는 mapping 테이블의 responseName을 검색할 부모를 의미한다.
2. Parent는 .으로 서브에 접근한다.

Ex) { ……., item:[{…}]}일때

item에서 responseName을 검색한다.

Parent가 null인경우 제일 최상위에서 찾는 것을 의미한다.(json)

### Mapping 테이블 스키마

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **테이블 명** | | Mapping | | |
| **COL 이름** | **데이터 타입** | | **PK 여부** | **NOT NULL 여부** |
| urlId | int | | Y | Y |
| colName | varchar(1024) | | Y | Y |
| responseName | varchar(1024) | |  | Y |
| type | int | |  | Y |

1. Phase는 0 -> 점점 커짐
2. HasChild : 0 -> child x, 2 -> child o
3. Type : 0 -> String, 1 -> int

### Response테이블 스키마

|  |  |
| --- | --- |
| **테이블 명** | Response{urlID} |

1. 테이블명은 Response뒤에 URL\_ID값을 붙여 고유한 값을 사용한다.
2. URL 및 테이블 생성 프로세스에서 입력한 컬럼명과 데이터 타입으로 테이블 생성
3. PK인 컬럼은 존재 하지 않는다.
4. 모든 컬럼에는 NOT NULL 옵션이 붙으면 안된다.

**\*추후 중복 값 확인은 모든 결과에 대해 and 연산을 실행**