이번엔 밸런스데이터와 세이브데이터를 분리해 SQL문이 너무 복잡해지지 않도록 한다

1. 빌딩

Table Building

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | Building\_id | 건물의 id |
| Text | Name | 건물의 이름 |
| Text | Image | 건물 이미지 파일 이름 |
| Text(30) | Description | 건물에 대한 설명 |
| Integer | Terrain\_id | 추가 이득을 볼 수 있는 지역 |

Table Function 이 부분을 좀 정규화시켜야될 필요가 있다

너무 다양한 데이터가 한곳에 모여있는 느낌

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | Function\_id | 기능의 id |
| Integer | Building\_id | 건물의 id |
| Integer | Resource\_id | 증가 자원 id |
| Integer | Value | 자원 증가량 |
| Integer | Add\_value | 이득 지역을 통한 추가량 |
| Number(1) | Type | 해당 기능의 종류  1. 패시브 : 매턴 반영  2. 건물 생성시 반영  3. On/Off 스킬  4. 일반기능 : 사용자가 버튼터치 |
| Number(1) | Lack\_type | 자원 증가량이 마이너스이다. 부족시 어떤 처리를 해야할까  1. 불가능  2. 경고 |

무기 업그레이드는 무기 레벨을 리소스에 두면 됨

Table BuildingView

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | view\_id | 리소스 표시뷰의 id |
| Integer | Building\_id | 건물의 id |
| Integer | Resource\_id | 보여줄 리소스 |

2. 자원

Table Resource

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | resource\_id | 자원의 id |
| Text | Name | 자원의 이름 |
| Number(2) | Type | 자원의 타입  1. 소모성  2. 식량  3. 비소모성 |

Table Maintain

한 자원을 유지하기 위해 소모되는 또 다른 자원

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | maintain\_id | 유지 항목 id |
| Integer | resource\_id | 유지해야하는 자원 |
| Integer | Consume \_id | 감소해야할 자원 |

사람/병사

빵/고기

Table Terrain

|  |  |  |
| --- | --- | --- |
| 데이터타입 | 이름 | 설명 |
| Integer primary key | terrain\_id | 지역 id |
| Text | Name | 지역 이름 |