**Couch DB**

**문서(json) 기반 데이터베이스**

-문서: real-world의 데이터가 저장된 형태이자 실제 데이터 구조 단위   
-미리 구현된 schema에 구애 받지 않는 자유로운 디자인 방식 제공

**CAP**

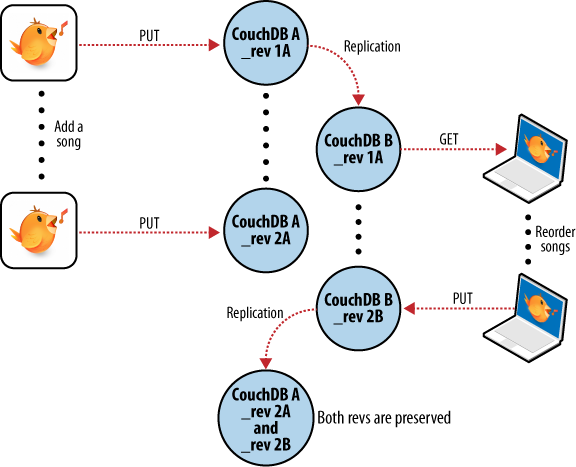
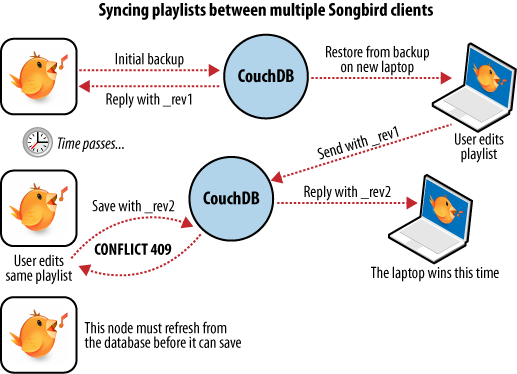
Consistency: 모든 클라이언트는 모두 같은 데이터를 사용 (업데이트 포함)  
Availability: 모든 클라이언트는 데이터의 읽기, 쓰기 및 여러 버전 접근이 가능  
Partition tolerance: 데이터베이스가 여러 서버에 물리적으로 분리되어서 작동

이중 AP를 만족.

- MVCC(Multi-Version Concurrency Control, 다중 동시성 제어): 버전 관리 시스템을 통해 문서를 버전 별로 관리하여 locking을 사용하지 않음

- \_rev의 자동 업데이트 기능으로 버전 관리 유용

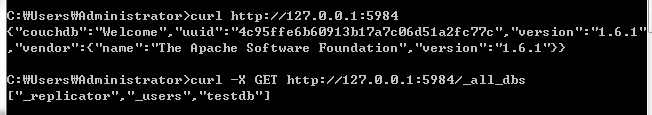
- Replication: 1:1, 1:N등 다양한 복제 기능을 제공. 지속적인 consistency 유지를 위해 서로 다른 데이터베이스끼리 계속해서 통신할 필요 없이 추가되거나 변경된 부분만을 복제, 동기화하여 eventual consistency를 유지



**HTTP기반 REST API 제공**

사용자가 HTTP를 이용하여 요청을 보내면 CouchDB도 HTTP를 통해 json문자열로 응답  
 \* curl 유틸리티 사용해서 원시적인 HTTP 명령 실행 / 자세한 작업 수행 과정 확인 가능

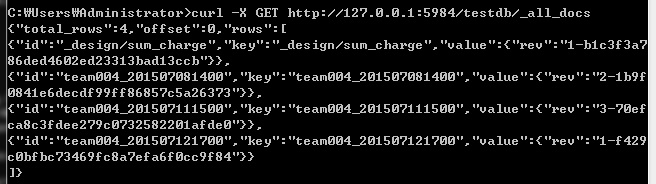
* DB 호출 및 목록 조회



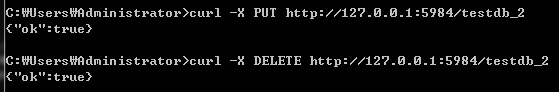
* DB 호출 요청-응답 과정



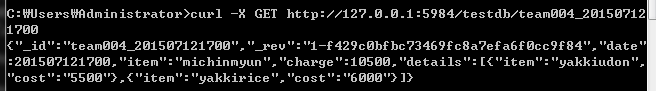
* ‘testdb’ 내 모든 문서 목록 조회



* ‘testdb\_2’ 생성 및 삭제

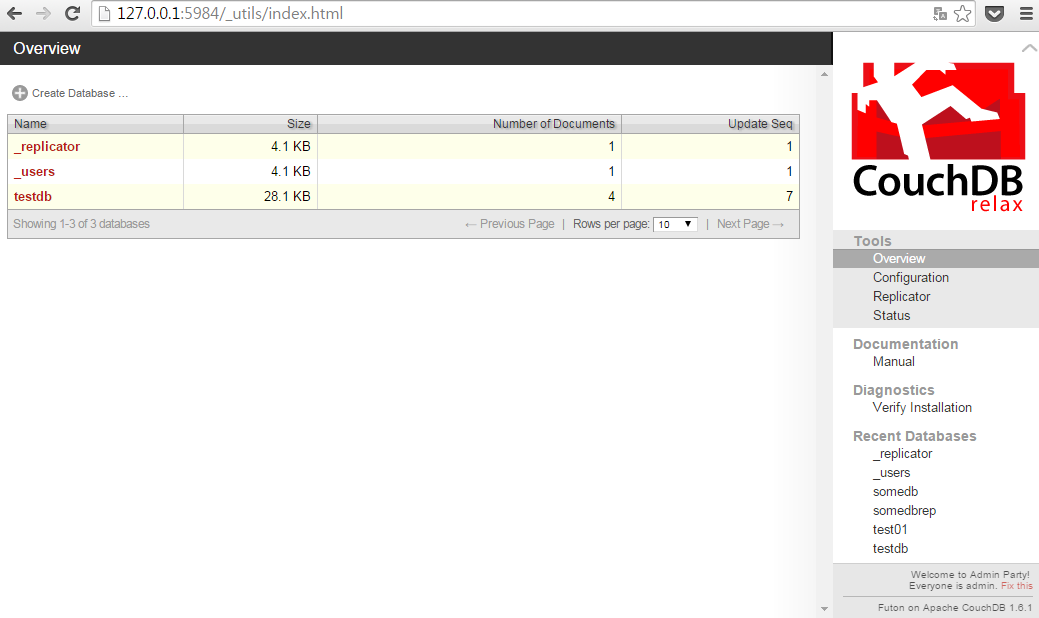


* ‘testdb’내의 id가 ‘team004\_201507121700’인 문서 상세 조회

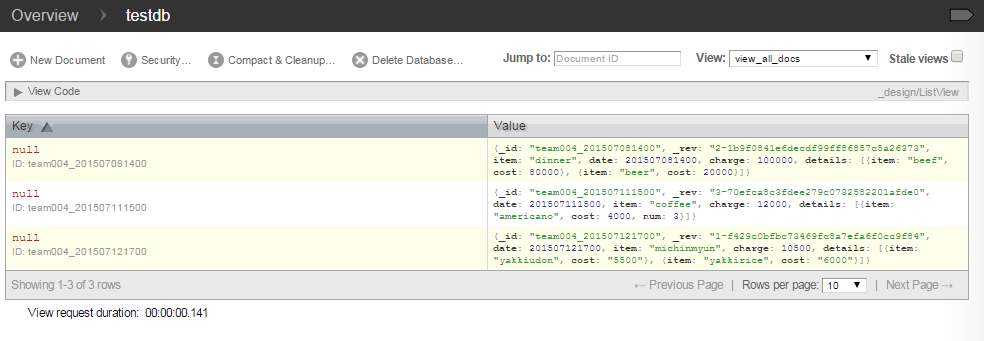


**Futon – 웹 기반 관리UI**

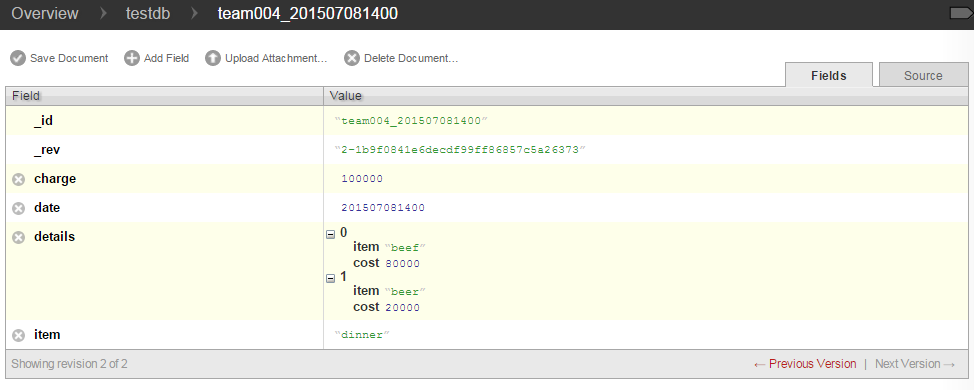
데이터베이스 생성 및 조회, 업데이트, 삭제, 사용자 관리 등 다양한 작업을 쉽게 수행할 수 있다



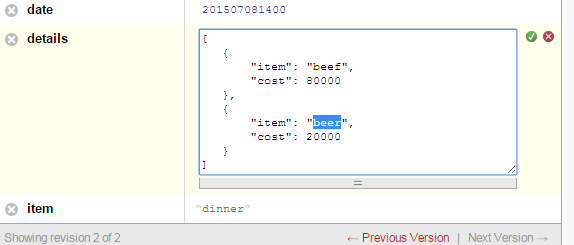
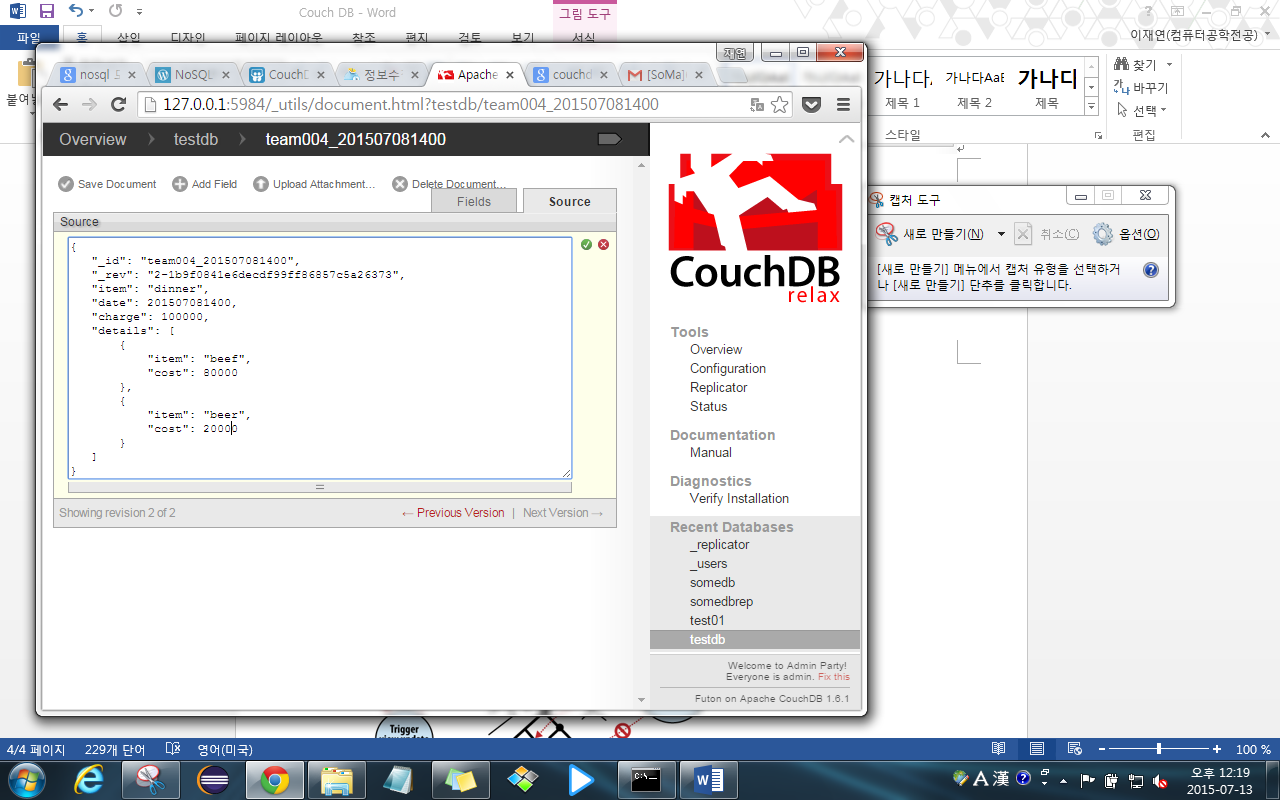
* testdb 내부 조회



* ‘team004\_201507081400’ 문서 조회

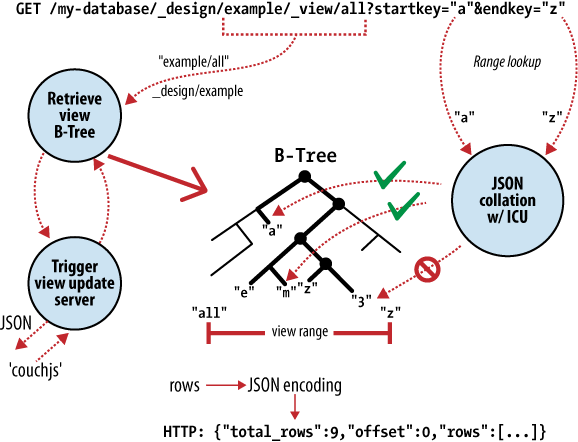


* Json 코드 입력 및 수정

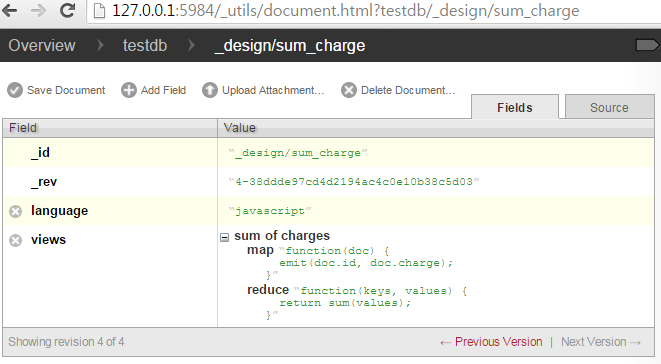
 

* ‘Save Document’선택시 \_rev가 자동으로 갱신. 하단의 Previous Version/Next Version을 통해 이전 버전 접근 용이함

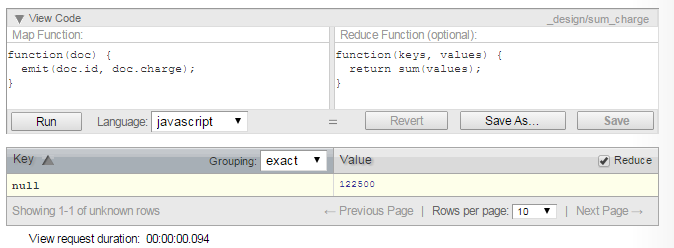
**Map/reduce 기능**



* View: 문서를 목적에 따라 특정한 순서대로 데이터를 추출하는 것. map, reduce 각 함수는 ‘view’로 결합되어 데이터베이스 내에 \_design/… 을 id로 갖는 design문서로 저장됨



* \_design/sum\_charge를 이용해 합계 계산 결과



* B-트리 엔진에 기반하여 문서와 뷰를 인덱싱, 빠르고 안정적인 속도 제공

\*\* For SOMA Expensify