**Database Cache Server Architecture**

|  |  |  |  |
| --- | --- | --- | --- |
| **리비전** | **날짜** | **작성자** | **내용** |
| 0.1 | 2010-12-06 | 김병수 | 초안 작성 |

1. 목적
   1. 응용 프로그램과 DBMS 중간에 위치하는 미들웨어(middle-ware)로, DBMS에 적재된 데이터를 캐싱(caching)하여 응용 프로그램에서 요구하는 데이터에 대한 접근 속도를 향상시킨다.
2. 기능
   1. 상대적으로 DB 부하가 높은 쿼리 결과를 메모리 상의 자료구조에 저장하여 직접적인 DB 사용 빈도를 낮춘다(예, 사용자, 캐릭터, 인벤토리 정보 등).
   2. 캐싱 대상이 되는 데이터는 DB에 직접적으로 반영
   3. 캐싱된 데이터는 일정 조건(시간, 갱신 빈도 또는 수동)에 따라 DB에 반영된다.
   4. 변경된 데이터는 최대한 모아서 DB에 반영해야 한다.
   5. 캐싱의 프로파일링(profiling) 정보를 로깅(logging)한다.
3. 구조

TODO:

1. 제약 사항
   1. 데이터 무결성: DB와 메모리 캐시 간에 데이터가 일치(동기화)해야 한다.
   2. 로그아웃한 특정 사용자의 데이터가 DB에 반영되지 않았다면, 반영될 때까지 로그인을 거부해야 한다.
2. 위험 요소
   1. 데이터 무결성이 위반될 경우의 에러 처리는 어떻게 할 것인가?
   2. 캐시 서버의 비정상 종료로 인해 DB에 반영되지 않은 데이터의 처리는 어떻게 할 것인가?
3. 관련 자료
   1. <http://www.gpgstudy.com/forum/viewtopic.php?topic=7243>
   2. <http://www.gpgstudy.com/forum/viewtopic.php?p=123698>
   3. <http://cafe.naver.com/ongameserver/4334>
   4. <http://cafe.naver.com/ongameserver/80>
4. 관련 아이디어
   1. 경매 시스템인 경우 일반 DB 보다 좀 더 효율적인 SQLite를 사용할 수 있다.
   2. Memcached
   3. [특허 출원](http://academic.naver.com/view.nhn?doc_id=1695785&ApplicationNumber=1020040108154&dir_id=0&page=0&query=%EA%B2%8C%EC%9E%84%20%EB%8D%B0%EC%9D%B4%ED%84%B0%EB%B2%A0%EC%9D%B4%EC%8A%A4%20%EC%BA%90%EC%8B%9C%20%EC%84%9C%EB%B2%84) -\_-;