Journal of The Korea Institute of Information Security & Cryptology(JKIISC) ISSN 1598-3986(Print)
VOL.23, NO.5, October 2013 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2013.23.5.947

Oracle 데이터베이스의 삭제된 레코드 복구 기법*

최 종 현,† 정 두 원, 이 상 진‡
고려대학교 정보보호대학원

The method of recovery for deleted record in Oracle Database*

Jong-Hyun Choi,† DooWon Jeong, Sangjin Lee‡
Center for Information Security Technologies(CIST), Korea University

요 약

기업의 정보는 대부분 데이터베이스에 보관된다. 따라서 기업의 범죄 행위를 조사하기 위해서는 데이터베이스에 대한 포렌식 분석이 중요하며 삭제 레코드 복구 기술을 개발할 필요가 있다. 이에 본 논문은 전 세계적으로 가장 많이 쓰이는 Oracle 데이터베이스의 테이블스페이스 파일 구조와 테이블정보를 저장하고 있는 시스템 테이블에 대해 분석하고, 이를 통해 Oracle 테이블스페이스에서 삭제된 레코드를 복구할 수 있는 방법을 제시한다.

ABSTRACT

Most of the enterprise information is stored in the database. Therefore, in order to investigate the company's criminal behavior, forensic analysis is important for the database and delete record is a need to develop recovery techniques. This paper is explained structure of the oracle database tablespace file and analyzed system tables that stored table information. Further, we suggests a method of recovery for deleted record in oracle tablespace.

Keywords: Oracle, Database, Deleted record, recovery, forensic

I. 서 론

일반적으로 조직은 방대한 양의 데이터를 통합적으로 관리하기 위해 데이터베이스를 이용하므로, 데이터베이스에는 업무데이터, 개인정보와 같은 중요한 정보가 저장되어 있는 경우가 많다. 이는 의미 있는 정보가 남아있을 가능성이 높다는 것을 뜻하므로 디지털 포렌식 관점에서 데이터베이스가 중요한 조사대상임을 알 수 있다.

데이터베이스에서 정상적인 레코드를 추출하는 것도 중요하지만, 삭제된 레코드를 복구하는 것도 중요하다. 삭제된 레코드 복구를 통해 사용자의 행위를 재구성할 수 있고, 용의자의 고의적 증거 인멸 행위에 대응할 수 있으므로 수사에 큰 도움을 줄 수 있다.

테이블스페이스는 데이터베이스에서 실제 데이터를 물리적으로 저장하는 공간을 말한다. 이 테이블스페이스를 분석하면 데이터베이스의 테이블 데이터와 테이블 스키마 정보를 얻을 수 있다.

IDC에 따르면 Oracle 데이터베이스는 전 세계적으로 40.7%의 점유율을 기록하고 있으며 국내 데이터베이스 시장에서 Oracle 데이터베이스는 60% 점유율을 기록하고 있다[1][2].

본 논문에서는 전 세계적으로 많이 쓰이고 있는 Oracle 데이터베이스 대상으로 테이블스페이스 파일 구조와 시스템 테이블에 대해서 알아본다. 이를 바탕으로 레코드 삭제 실험, 레코드 삽입 실험, 테이블 삭

접수일(2013년 8월 29일), 수정일(1차: 2013년 9월 27일, 2차: 2013년 10월 4일), 게재확정일(2013년 10월 4일)
* 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단-공공복지안전사업의 지원을 받아 수행된 연구임(2012M3A2A1051106)
† 주저자, antares0531@gmail.com
‡ 교신저자, sangjin@korea.ac.kr(Corresponding author)

제 실험, 데이터베이스 최적화 실험을 하였으며, Oracle 테이블스페이스에서 삭제된 레코드 복구 기법을 제시하고자 한다. 본 논문의 실험 대상은 Windows 환경에서 구동한 Oracle 9i, 10g, 11g 이다.

II. 관련 연구

현재까지의 데이터베이스 포렌식에 대한 선행 연구는 트랜잭션 로그를 토대로 레코드를 복구하는 것에 집중되어 있다. P. Wright는 Oracle 데이터베이스에서 redolog와 LogMiner를 통해 데이터베이스의 변경사항을 추적하는 방안과 이전 시점의 데이터를 복원하는 방법에 대해서 설명하였다[3]. 또한 Heloise Pieterse는 데이터베이스에서 암호화, 분할 저장 등을 이용하여 데이터 값이나 레코드, 테이블을 은닉하는 기법에 대해서 정리하였으며, 트랜잭션 로그의 중요성에 대해서 강조하였다[4]. Oluwasola Mary는 데이터베이스에서 쿼리를 대수학으로 표현할 수 있음을 보였다. 그리고 쿼리를 역으로도 나타낼 수 있으며, 이 수식을 저장하는 Relational Algebra Log를 제안하였다. 이 로그를 통해 이전 시점의 데이터들을 복원하는 기법에 대해서 설명하였다[5].

일반적으로 Oracle 데이터베이스의 삭제된 레코드를 복원할 때는 DBF파일과 트랜잭션 로그가 쌍으로 존재해야한다. 그리고 운용중인 Oracle 데이터베이스에 LogMiner를 이용해 삭제된 레코드를 복원한다. 하지만 이 방법은 트랜잭션 로그가 기록된 시점만 복구가 가능하며, 기록되지 않은 시점은 복구하지 못한다. 또한 Oracle 데이터베이스에서 트랜잭션 로그를 남기는 방법 중 No Archive Mode의 경우 미리 지정한 용량을 초과하게 되면 기존의 트랜잭션 로그에 덮어쓰기 때문에 Oracle 데이터베이스 운용 환경이나 회사 정책에 따라 복구 여부가 결정될 수도 있다[6]. 이에 트랜잭션 로그에 비종속적으로 데이터베이스에서 삭제된 레코드를 복구할 수 있는 방법이 필요하다.

본 논문에서는 선행 연구와 다르게 트랜잭션 로그가 아닌 Oracle 테이블스페이스 파일인 DBF파일을 대상으로 파일 내의 잔존하는 데이터를 파싱하여 삭제된 레코드를 복구할 수 있는 기법에 대해서 제안하고자 한다. 본 논문은 Oracle 버전 중 9i, 10g, 11g를 대상으로 하였으며, Windows 환경에서 실험하였다. 또한 Oracle 데이터베이스에서는 테이블스페이스를 암호화가 가능한데 이를 하지 않았다고 가정한다.

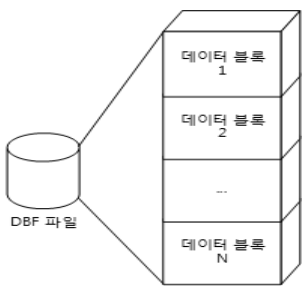
III. Oracle 테이블스페이스 구조

Oracle 데이터베이스에서 삭제된 레코드를 복구하기 위해서는 테이블스페이스 파일 구조에 대해서 알아야 한다. Oracle 테이블스페이스는 TEMP, USER, SYSAUX, SYSTEM, EXAMPLE, UNDOTBS가 있으며, 각 테이블스페이스의 역할은 [표 1]과 같다.

[표 1] Oracle 테이블스페이스 설명

테이블스페이스 명	설명
TEMP	SQL 처리시 요구되는 임시 테이블과 인덱스 저장
USER	사용자가 생성한 객체 저장
SYSAUX	시스템 정보 저장
SYSTEM	데이터 딕셔너리 저장
EXAMPLE	샘플 스키마 저장
UNDOTBS	Undo 정보 저장

이 중에서 테이블과 레코드들을 저장하고 있는 테이블스페이스는 SYSTEM이다.



[그림 1] DBF 파일 구성

테이블스페이스는 [그림 1]처럼 데이터 블록으로 구성되어 있다. 데이터 블록은 데이터베이스가 사용하는 가장 작은 저장 단위로써 데이터베이스 안에 있는 데이터를 읽거나 쓸 때 사용되는 조작 단위이다. 데이터 블록의 크기는 Oracle을 설치할 때 설정할 수 있으며, 사용 중에는 변경할 수 없다.

그리고 해당 시스템에서 블록 사이즈를 확인할 수 있는데, 이는 데이터베이스 생성 로그를 통해 확인가능하다. 기본 경로는 Oracle 데이터베이스가 설치된 곳에서 \admin\[데이터베이스명]\bdump\alert_데이터베이스명.log에서 확인가능하다. [그림 2]는 해당 log를 나타낸 것이며 db_block_size항목을 보

```
Starting up ORACLE RDBMS Version: 10.2.0.1.0.
System parameters with non-default values:
  processes                = 150
  nls_language             = KOREAN
  nls_territory            = KOREA
  sga_target               = 612368384
  control_files            = C:₩ORACLE₩PRODUCT₩10.2.0₩ORADATA₩DFRC₩CONTROL01
  db_block_size            = 8192
  compatible               = 10.2.0.1.0
  db_file_multiblock_read_count= 16
  db_recovery_file_dest    = C:₩oracle₩product₩10.2.0₩flash_recovery_area
  db_recovery_file_dest_size= 2147483648
  undo_management          = AUTO
  undo_tablespace          = UNDOTBS1
  remote_login_passwordfile= EXCLUSIVE
  db_domain                =
  dispatchers              = (PROTOCOL=TCP) (SERVICE=DFRCXDB)
  job_queue_processes      = 10
  audit_file_dest          = C:₩ORACLE₩PRODUCT₩10.2.0₩ADMIN₩DFRC₩ADUMP
  background_dump_dest     = C:₩ORACLE₩PRODUCT₩10.2.0₩ADMIN₩DFRC₩BDUMP
  user_dump_dest           = C:₩ORACLE₩PRODUCT₩10.2.0₩ADMIN₩DFRC₩UDUMP
  core_dump_dest           = C:₩ORACLE₩PRODUCT₩10.2.0₩ADMIN₩DFRC₩CDUMP
  db_name                  = DFRC
  open_cursors             = 300
  pga_aggregate_target     = 203423744
```

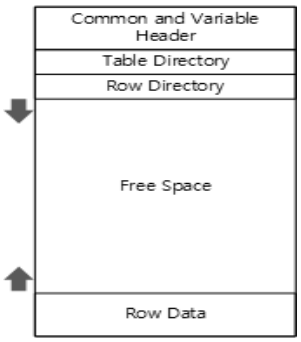
〔그림 2〕 alert_데이터베이스명.log 파일

면 데이터 블록 크기가 8192라는 것을 알 수 있다.

이 절에서는 테이블스페이스를 구성하고 있는 데이터 블록의 구조와 테이블들의 주요 정보를 알 수 있는 시스템테이블에 대해서 알아본다.

3.1 데이터 블록 구조

Oracle 데이터 블록은 Common and Variable Header, Table Directory, Row Directory, Free Space, Row Data로 구성된다[7]. 〔그림 3〕은 데이터 블록의 구성을 나타낸 것이다.



〔그림 3〕 데이터 블록 구조

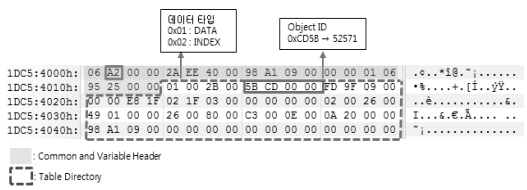
데이터 블록에서 레코드가 증가할수록 Row Directory와 Row Data도 증가한다. Row Directory는 데이터 블록 위에서부터 데이터를 저장하며 Row Data는 데이터 블록 아래에서부터 데이터를 저장한다. 각 구성요소별로 획득할 수 있는 정보는 〔표 2〕와 같다.

〔그림 4〕는 Common and Variable Header와 Table Directory의 화면이다. Common and Variable Header의 크기는 20바이트이며 Oracle

〔표 2〕 데이터 블록에서 획득할 수 있는 정보

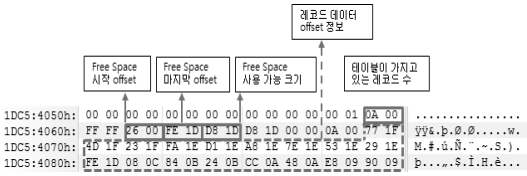
구성요소	설명
Common and Variable Header	Oracle 버전정보
Table Directory	데이터 블록 타입, Object ID
Row Directory	레코드 개수, Free Space 크기, 레코드의 offset 정보
Free Space	Free Space 크기
Row Data	데이터

버전정보를 알 수 있다. Table Directory에서는 해당 데이터 블록의 타입을 알 수 있는데 0x01이면 DATA, 0x02이면 INDEX를 저장하고 있다는 것을 뜻한다. 또한 Object ID를 알 수 있는데 이는 Oracle 테이블스페이스에서 객체마다 부여하는 것으로 테이블을 식별할 때 사용한다. 〔그림 4〕의 테이블은 0xCD5B의 값을 가지므로 Object ID가 52571이라는 것을 알 수 있다.



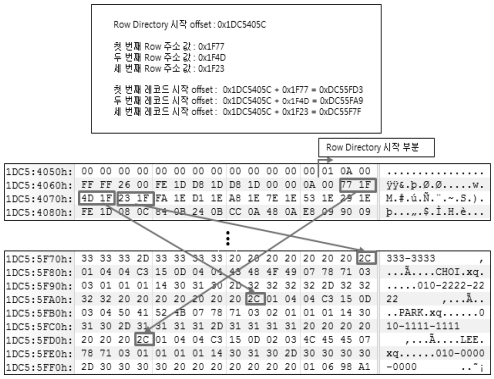
〔그림 4〕 데이터 블록 헤더

Row Directory에서는 해당 테이블이 가지고 있는 레코드 개수와 Free Space 크기, 레코드의 offset 정보를 얻을 수 있다. 〔그림 5〕는 Row Directory를 캡쳐한 화면이다.



〔그림 5〕 Row Directory

Row Directory에는 레코드 데이터 offset 정보가 있다. 이는 Row Directory 시작 offset으로부터의 거리를 뜻한다. 〔그림 5〕에서 Row Directory 시작 offset은 0x1DC5405C가 되고, 첫 번째 값은 0x1F77이다. 이 둘을 더하게 되면 0xDC55FD3이 나오고, 이는 첫 번째 레코드의 시작 위치를 뜻한다.



〔그림 6〕 Row Directory에서의 레코드 데이터 추적

이와 같은 계산법으로 각 레코드의 위치를 계산해보면 〔그림 6〕과 같이 레코드를 추적할 수 있다.

3.2 시스템 테이블

Oracle 데이터베이스에는 테이블 정보나 테이블 컬럼 정보, 환경 설정 등을 저장하고 있는 시스템 테이블이 있다. 이 테이블도 SYSTEM.DBF파일에 저장되어 있다. 각 시스템 테이블은 고유한 Object ID를 가지고 있다. 따라서 시스템 테이블의 Object ID를 알고 있으면 테이블의 이름과 테이블의 컬럼명 등 스키마 정보를 획득할 수 있다.

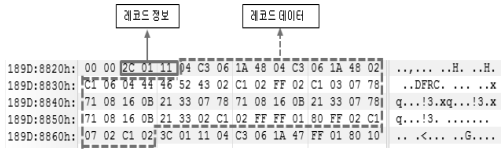
시스템 테이블 중 OBJ$와 C_OBJ#은 테이블의 스키마 정보를 획득할 수 있다. OBJ$에서는 테이블명을 얻을 수 있으며 C_OBJ#에서는 테이블 컬럼 명과 데이터형, 데이터 길이정보를 얻을 수 있다. 그리고 OBJ$와 C_OBJ#은 고유한 Object ID를 가지고 있다. 이는 〔표 3〕과 같다.

〔표 3〕 C_OBJ#와 OBJ$의 Object ID

테이블 명	Object ID
C_OBJ#	2
OBJ$	18

OBJ$ 테이블의 스키마 정보는 Oracle 공식홈페이지에서 찾을 수 있다[8]. 이 테이블에서는 테이블명, Object ID, 테이블 생성 시간을 알 수 있다. 해당 테이블에서 row data영역의 한 레코드를 살펴보면 〔그림 7〕과 같다.

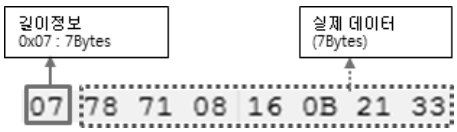
row data의 레코드 정보영역에서 맨 앞 3바이트는 레코드 정보를 나타낸다. 첫 바이트는 레코드의 상



〔그림 7〕 OBJ$의 row data 영역 레코드

태 플래그이며 세 번째 바이트는 컬럼의 개수를 뜻한다. 〔그림 7〕에서 세 번째 바이트는 0x11이므로 17개의 컬럼을 가지고 있음을 뜻한다.

또한 레코드 데이터영역에서는 실제 레코드 데이터들이 저장되어 있다. Oracle에서 지원하는 자료형마다 저장하는 방식이 다르므로 유의해야 한다[9][10]. 기본적인 구조는 〔그림 8〕과 같다. 맨 첫 번째 바이트가 해당 컬럼의 길이 정보를 뜻한다. 〔그림 8〕에서는 첫 번째 바이트가 0x07이므로 7Bytes의 데이터를 저장하고 있다는 것을 뜻한다.



〔그림 8〕 데이터 저장 방식

Oracle 자료형 중 DATE 자료형은 날짜와 시간을 고정 길이로 표현하는데 사용하며, 7 Bytes를 사용한다. DATE 자료형은 세기, 년, 월, 일, 시, 분, 초를 1Byte로 저장하고 있다. 자세한 내용은 〔그림 9〕에서 볼 수 있다. 또한 NUMBER 자료형은 가변 길이의 숫자 데이터를 저장하는데 사용하며, 1바이트에 2자리가 저장되는 100진수를 사용하고 있다.

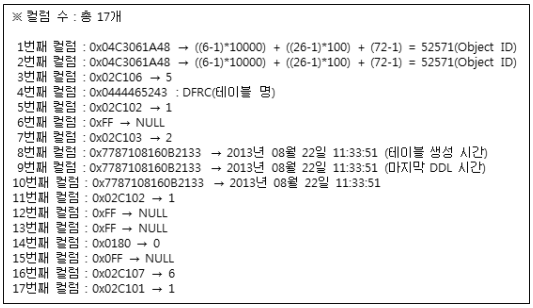
※ DATE 자료형 예시

07 78 71 08 16 0B 21 33
① ② ③ ④ ⑤ ⑥ ⑦ ⑧

① 0x07 : 길이정보(7Bytes)
② 0x78 : 120 – 100 = 20세기
③ 0x71 : 113 – 100 = 13년
④ 0x08 : 08월
⑤ 0x16 : 22일
⑥ 0x0B : 11시
⑦ 0x21 : 33분
⑧ 0x33 : 51초

=> 2013년 08월 22일 11시 33분 51초를 나타냄

〔그림 9〕 DATE 자료형 저장 방식



〔그림 10〕 OBJ$에서 획득한 레코드 정보

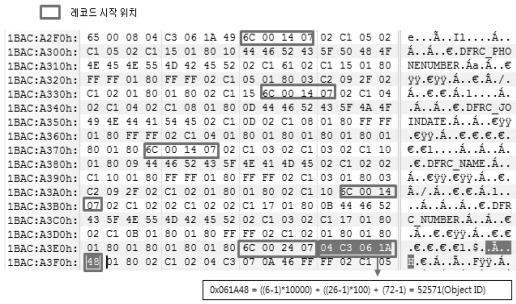
이와 같은 방법으로 〔그림 7〕의 레코드 데이터는 맨 앞의 레코드 길이 정보를 가지고 컬럼 정보를 획득할 수 있는데, 획득한 정보를 정리하면 〔그림 10〕과 같다.

〔그림 10〕을 보면 OBJ$ 테이블에서 한 레코드를 분석해 테이블명이 DFRC라는 것과 테이블 생성시간, Object ID를 알 수 있다.

C_OBJ#는 시스템 클러스터이다. 클러스터란 테이블에서 쓰이는 컬럼 정보들을 하나의 그룹으로 묶어 저장하는 오라클 데이터베이스 객체이다. 이 클러스터에는 테이블의 Object ID와, 컬럼 명, 컬럼 자료형 타입, 크기정보 등 스키마 정보를 획득할 수 있다. 〔그림 11〕은 C_OBJ#에서 한 클러스터를 캡쳐한 화면이다. 각 레코드 시작 부분을 표시해 두었으며 제일 하단의 블록 부분을 보면 Object ID를 볼 수 있다. 또한 클러스터에 있는 레코드들의 앞부분을 보면 4번째 바이트가 똑같은 것을 볼 수 있는데 이는 클러스터의 번호를 뜻한다.

C_OBJ# 주요 컬럼은 〔표 4〕에 정리하였다. C_OBJ#의 6번째 컬럼은 자료형 타입을 뜻하며, 각 자료형의 고유한 값은 〔표 5〕에 정리하였다.

앞서 언급한 OBJ$ 테이블, OBJ# 클러스터, 테이



〔그림 11〕 C_OBJ#의 클러스터

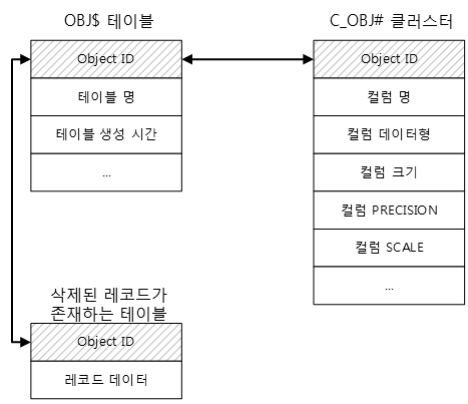
〔표 4〕 C_OBJ# 주요 컬럼

컬럼 순서	설명
1	컬럼 순서
5	컬럼 명
6	자료형 타입
7	자료형 길이 정보
9	숫자 데이터의 PRECISION
10	숫자 데이터의 SCALE

〔표 5〕 자료형 타입

값	자료형 타입
1	VARCHAR2 / NVARCHAR2
2	NUMBER
8	LONG
12	DATE
23	RAW
24	LONG RAW
69	ROWID
96	CHAR / NCHAR
112	CLOB / NCLOB
113	BLOB
114	BFILE
181	TIMESTAMP
182	INTERVAL
183	INTERVAL
208	UROWID
231	TIMESTAMP

블에는 모두 Object ID가 들어 있다. 이를 가지고 테이블 스키마 정보와 테이블 레코드를 얻을 수 있다. 〔그림 12〕는 이들의 관계도를 나타낸 것이다.

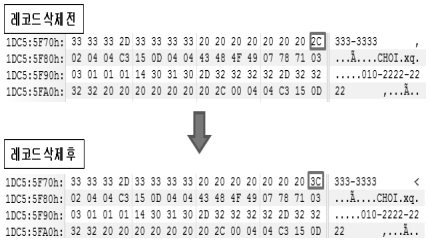


〔그림 12〕 OBJ$, C_OBJ#, 사용자테이블의 관계도

IV. 테이블스페이스의 삭제된 레코드 복구 기법

4.1 레코드 삭제 실험

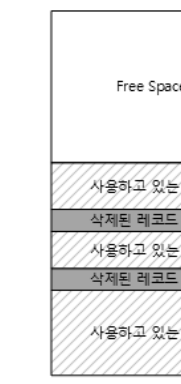
Oracle 데이터베이스에서 레코드를 삭제하게 되면 어떠한 데이터가 변하는지 레코드 삭제 전과 삭제 후로 나누어 비교를 해 보았다. 실험 방법은 다음과 같다. Oracle 데이터베이스에서 DFRC라는 테이블을 생성한 뒤, 10개의 컬럼을 삽입하였다. 그 다음 DFRC_NAME이 CHOI라는 레코드를 DELETE 쿼리문을 통해 삭제하였다. 이 때 OBJ$ 테이블과 C_OBJ#에는 변화가 없었으며 해당 레코드를 저장하고 있는 데이터 블록에만 변화가 생겼다. row data에서 레코드 정보를 나타내는 부분이 0x2C에서 0x3C로 바뀌었다. 이것은 〔그림 13〕을 통해 확인할 수 있다.



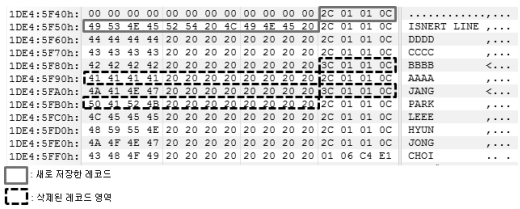
〔그림 13〕 레코드 삭제 후 row data의 변화

4.2 레코드 삽입 실험

한 테이블에서 레코드를 삭제했을 때 데이터 블록에는 〔그림 14〕처럼 레코드가 삭제된 영역이 생긴다. 이 실험은 〔그림 14〕와 같은 상황에서 INSERT 쿼리문을 통해 새로운 레코드를 저장할 때 Free Space 영역에 저장하는지 아니면 삭제된 레코드 영역에 덮어써지는지 알아보려고 것이다. 정확한 실험을 위해 레코드의 길이는 똑같이 하였다. 실험 결과 〔그림 15〕에서처럼 Free Space영역에 저장하였다. 이 실험을 통해 새로운 레코드를 저장할 때 삭제된 레코드 영역이 아니라 Free Space영역에 저장하는 것을 알 수 있다.



〔그림 14〕 삭제된 레코드가 있는 테이블 영역



〔그림 15〕 레코드 삽입 실험 결과

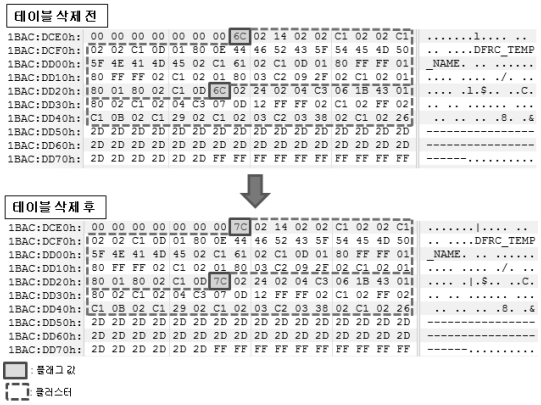
4.3 테이블 삭제 실험

Oracle 데이터베이스에서 테이블을 삭제하게 되면 OBJ$ 테이블, C_OBJ# 클러스터, 데이터 블록에 어떠한 데이터가 변하는지 테이블 삭제 전과 삭제 후로 나누어 비교해 보았다. 테이블을 삭제할 때는 DROP 쿼리문 사용하였다. OBJ$ 테이블에서 〔그림 16〕처럼 플래그 값이 0x2C에서 0x3C로 바뀌었다.

〔그림 16〕 테이블 삭제 시 OBJ$의 변화

그리고 C_OBJ# 클러스터에는 〔그림 17〕처럼 클러스터들의 플래그 값이 0x6C에서 0x7C로 바뀌었다.

삭제한 테이블의 데이터 블록에는 데이터의 변화는 없었다. 하지만 테이블스페이스의 비할당영역으로 분류되어 다른 데이터 블록에 의해 덮어써질 수 있다. 이 실험을 통해 테이블을 삭제할 시 OBJ$와 C_OBJ#에서 삭제한 테이블 명과 스키마 정보를 할 수 있었으며, 데이터 블록이 덮어써져 있지 않으면 복

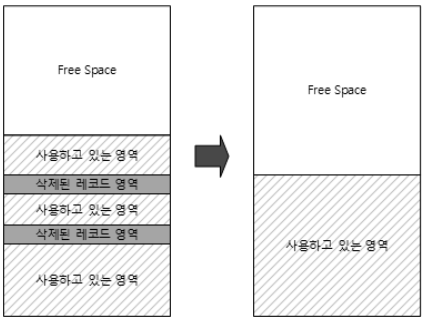


〔그림 17〕 테이블 삭제 시 C_OBJ#의 변화

구가 가능하다는 것을 알 수 있었다.

4.4 데이터베이스 최적화 실험

다른 실험들을 통해 데이터베이스에서 테이블이나 레코드들을 삭제할 때 플래그 값만 바뀌며 실질적인 데이터는 남아있는 것을 확인할 수 있었다. 이렇게 되면 DBF파일을 효율적으로 사용하지 못한다. 이에 Oracle에서는 10g버전부터 Shrink라는 기능을 추가하였다. Shrink는 테이블이 실질적으로 안 쓰는 부분을 정리해 주는 기능이다. 〔그림 18〕처럼 사용하지 않은 영역들을 없애 데이터베이스 공간 활용도를 높인다.



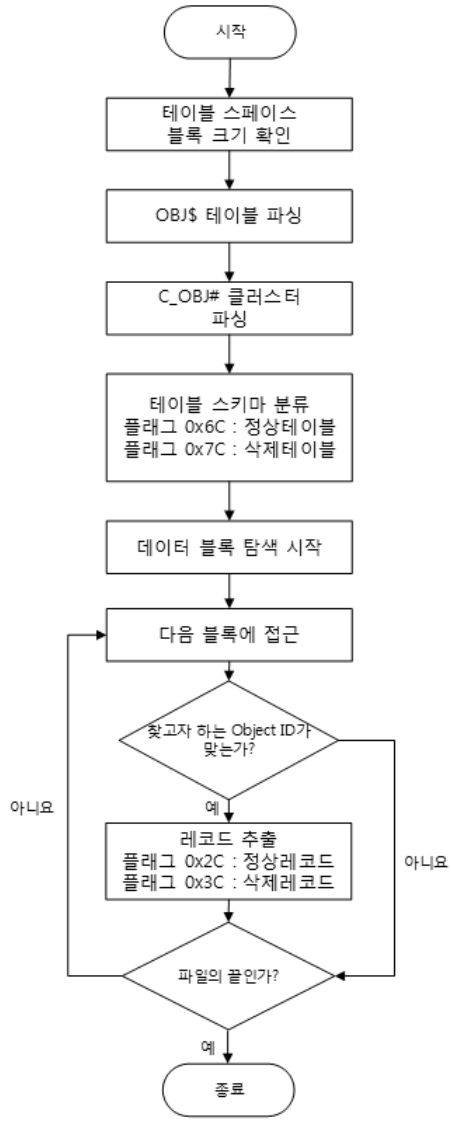
〔그림 18〕 Shrink 전(좌)와 Shrink 후(우)

이 경우 덮어써진 영역에 대해서는 복구를 할 수 없으며, Free Space에서 남아있는 레코드만 복구가 가능하다.

4.5 삭제된 레코드 복구 기법

삭제된 레코드를 복구하기 위해서는 먼저 테이블스페이스의 블록크기를 알아야 한다. 블록크기는 설치 로그파일에서 찾을 수 있다. 만약 로그파일이 없다면 해당 테이블스페이스에서 Oracle 버전을 나타내는 Oracle Type을 보고 블록 크기를 찾아야 한다.

그 다음 OBJ$ 테이블을 파싱한다. 이 과정에서 테이블 명과 Object ID를 알 수 있다.이때 플래그 값이



〔그림 19〕 Oracle 테이블스페이스에서의 삭제된 레코드 복구 기법

0x2C일 경우 정상적인 테이블이며 0x3C일 경우 삭제된 테이블을 뜻한다.

다음으로 C_OBJ# 클러스터를 파싱한다. 이 과정에서 테이블의 Object ID와 스키마 정보들을 알 수 있다. 이때 플래그 값이 0x6C일 경우 정상적인 테이블의 스키마 정보이며 0x7C일 경우 삭제된 테이블의 스키마 정보이다.

그리고 획득한 스키마 정보를 바탕으로 Object ID를 비교하면서 블록들을 탐색한다. 블록을 탐색할 때는 레코드 정보에서 첫 번째 플래그 정보를 읽는다. 이 플래그가 0x3C이면 삭제된 레코드임을 식별하고 복구할 수 있다. 이를 정리하면 〔그림 19〕와 같다.

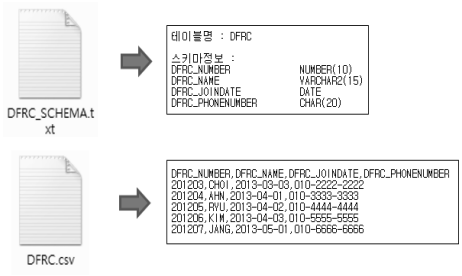
V. 실 험

4절에서 기술된 복구 기법을 토대로 삭제된 레코드를 복구하는 도구를 구현하였다. OBJ$를 통해 Oracle 테이블스페이스내의 모든 테이블에 대한 Object ID를 수집하고 이 수집된 Object ID를 통해 C_OBJ# 클러스터에서 컬럼 명을 수집한다. 이 과정을 통해 정상적인 테이블의 스키마 정보와 삭제된 테이블의 스키마 정보를 알 수 있다.

그 다음 수집된 Object ID를 가지고 있는 블록에서 삭제된 레코드를 복구한다. 테이블의 스키마 정보와 테이블의 삭제된 레코드 정보는 CSV 파일로 저장하며 각각 다른 파일에 저장된다.

실험 결과 Oracle 데이터베이스의 Shrink 기능을 사용하지 않았을 때에는 삭제된 레코드는 모두 복원할 수 있었다. 또한 삭제된 테이블은 스키마 정보는 모두 복원할 수 있었지만 레코드들은 데이터 블록이 덮어써지지 않은 경우에만 레코드들을 복원할 수 있었다. Oracle 데이터베이스에서 Shrink를 사용한 경우 Free Space에서 레코드들을 조각을 가지고 삭제된 레코드들을 일부 복원할 수 있었지만 대부분의 삭제된 레코드들은 덮어써져 복원을 할 수 없었다.

〔그림 20〕는 테이블의 스키마 정보와 테이블의 삭제된 레코드 정보를 추출해 낸 것이다.



〔그림 20〕 삭제된 레코드 복구 실험 결과

VI. 결 론

데이터베이스에는 중요한 정보가 저장되어 있어 디지털 포렌식 관점에서 의미 있는 정보가 있을 확률이 높다. 데이터베이스에서 정상적인 레코드를 추출하는 것도 중요하지만 삭제된 레코드를 복구하는 것도 중요하다. 현재 데이터베이스 데이터 복원에 대한 연구는 트랜잭션 로그를 기반으로 복원한다. 하지만 이 방법은 트랜잭션 로그가 존재하지 않거나 트랜잭션 로그가 기록한 이전시점의 데이터를 복원하고 싶은 경우에는 사용할 수 없다. 이에 트랜잭션 로그의 비종속적으로 데이터베이스의 삭제된 레코드들을 복원하는 방법이 필요하다.

이에 본 논문은 전 세계적으로 많이 쓰이고 있는 Oracle 데이터베이스 9i, 10g, 11g를 대상으로 테이블스페이스에서 OBJ$와 C_OBJ$를 통해 삭제된 테이블과 삭제된 레코드들을 복구하는 기법을 제안하였다. 또한 실험을 통해 삭제된 테이블과 삭제된 레코드가 정상적으로 복구되는 것을 확인하였다. 데이터베이스의 포렌식 조사를 진행함에 있어 트랜잭션 로그파일이 존재하지 않거나 트랜잭션 로그파일이 기록하지 않은 시점의 레코드들을 복구할 경우 본 논문에서 제시한 방법이 의미 있는 정보를 추출하는데 많은 도움이 될 것이다.

참고문헌

〔1〕 Oracle 전 세계 점유율, “http://apacmediacentre.oracle.com/content/detail.aspx?ReleaseID=6129&NewsAreaId=2”, Apr. 2013.

〔2〕 Oracle 국내 점유율, “http://www. dt.co.kr/contents.html?article_no=2013012302011060746002”, 2013년 1월.

〔3〕 SANS Computer Forensics, “Oracle Database Forensics using LogMine r,”, Jan. 2005.

〔4〕 Heloise Pieterse and Martin Olivier, “Data Hiding Techniques for Database

Environments," Advances in Digital Forensics Ⅷ, Vol. 383, pp. 289-301, Jan. 2012.
〔5〕 Oluwasola Mary Fasan and Matin Olivier, "Reconstruction in database forensics," Advances in Digital Forensics Ⅷ, Vol. 383, pp. 273-287, Jan. 2012.
〔6〕 Oracle 트랜잭션 로그 저장 모드, "http://docs.oracle.com/cd/B19306_01/server.102/b14231/archredo.htm"
〔7〕 데이터 블록 Format, "http://docs.oracle.com/cd/B28359_01/server.111/b28318/logical.htm"
〔8〕 OBJ$ 스키마 정보, "http://docs.oracle.com/cd/B1 4117_01/server.101/b10755/statviews_1102.htm#i1583352"
〔9〕 Oracle Data Types, "http://docs.oracle.com/cd/B28 359_01/server.111/b28318/datatype.htm"
〔10〕 Oracle, Oracle 9i Database: Data Types and Storage Internals, May. 2002.

〈저자소개〉

최 종 현 (Jong-Hyun Choi) 정회원
2012년 2월: 경희대학교 전자정보대학 컴퓨터공학과 공학사
2012년 3월~현재: 고려대학교 정보보호대학원 정보보호학과 석사과정
〈관심분야〉 디지털 포렌식

정 두 원 (Doo-won Jeong) 정회원
2011년 8월: 고려대학교 공과대학 산업경영공학과 공학사
2011년 9월~현재: 고려대학교 정보보호대학원 정보보호학과 석·박사통합과정
〈관심분야〉 디지털 포렌식, 정보보호, 빅데이터 분석

이 상 진 (Sang-jin Lee) 종신회원
1989년 2월~1999년 2월: 한국전자통신연구원 선임 연구원
1999년 2월~2001년 8월: 고려대학교 자연과학대학 조교수
2001년 9월~현재: 고려대학교 정보보호대학원 교수
〈관심분야〉 대칭키 암호, 정보은닉이론, 디지털 포렌식