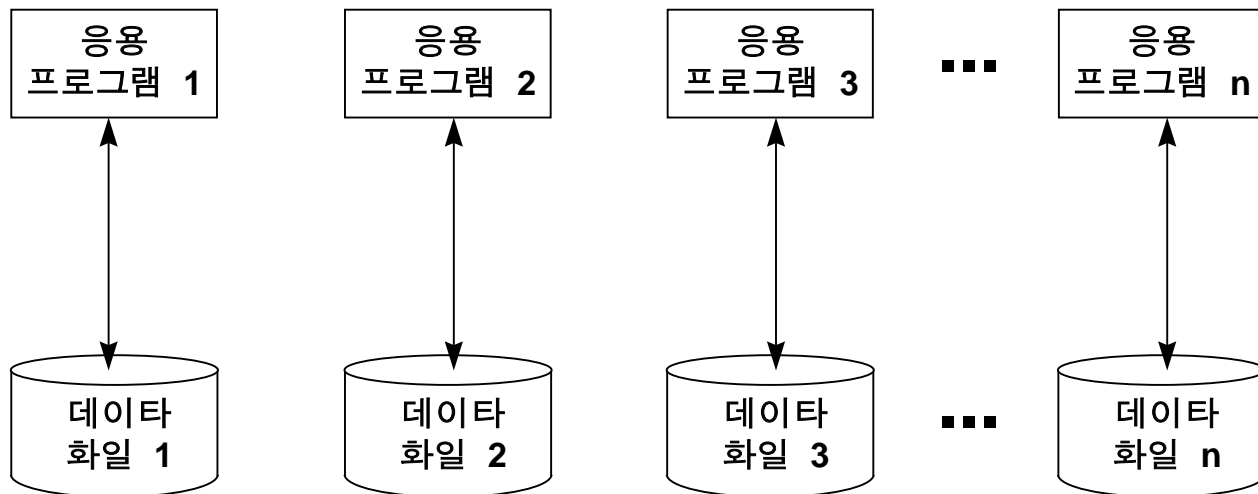


14. 데이터베이스

❖ 파일과 데이터베이스

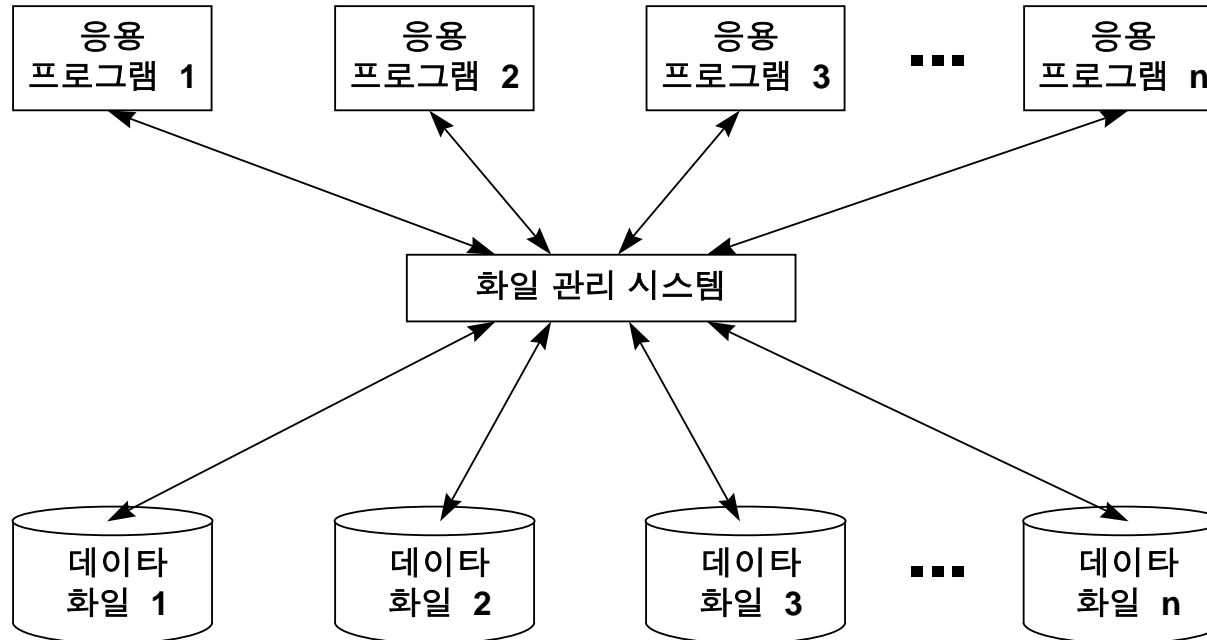
◆ 개별 파일 처리 시스템

- 각 응용 프로그램이 개별적인 파일 처리 루틴 작성
- 응용 프로그래머에 부담



▶ 파일 관리 시스템

- 파일 처리 루틴 공유
- 데이터 파일의 공유는 없음



▶ 데이터의 중복성(data redundancy)

- 급여 화일

| | | | | | | | | |
|------|----|----|----|----|-----|-----|--------|----|
| 교수번호 | 이름 | 학과 | 호봉 | 봉급 | 공제액 | 지급액 | 주민등록번호 | 주소 |
|------|----|----|----|----|-----|-----|--------|----|

- 인사 화일

| | | | | | | | |
|------|----|----|----|--------|-------|----|----|
| 교수번호 | 이름 | 학과 | 호봉 | 주민등록번호 | 연구실번호 | 주소 | 경력 |
|------|----|----|----|--------|-------|----|----|

==> 데이터의 중복, 모순성 야기.

▶ 데이터 종속성(data dependency)

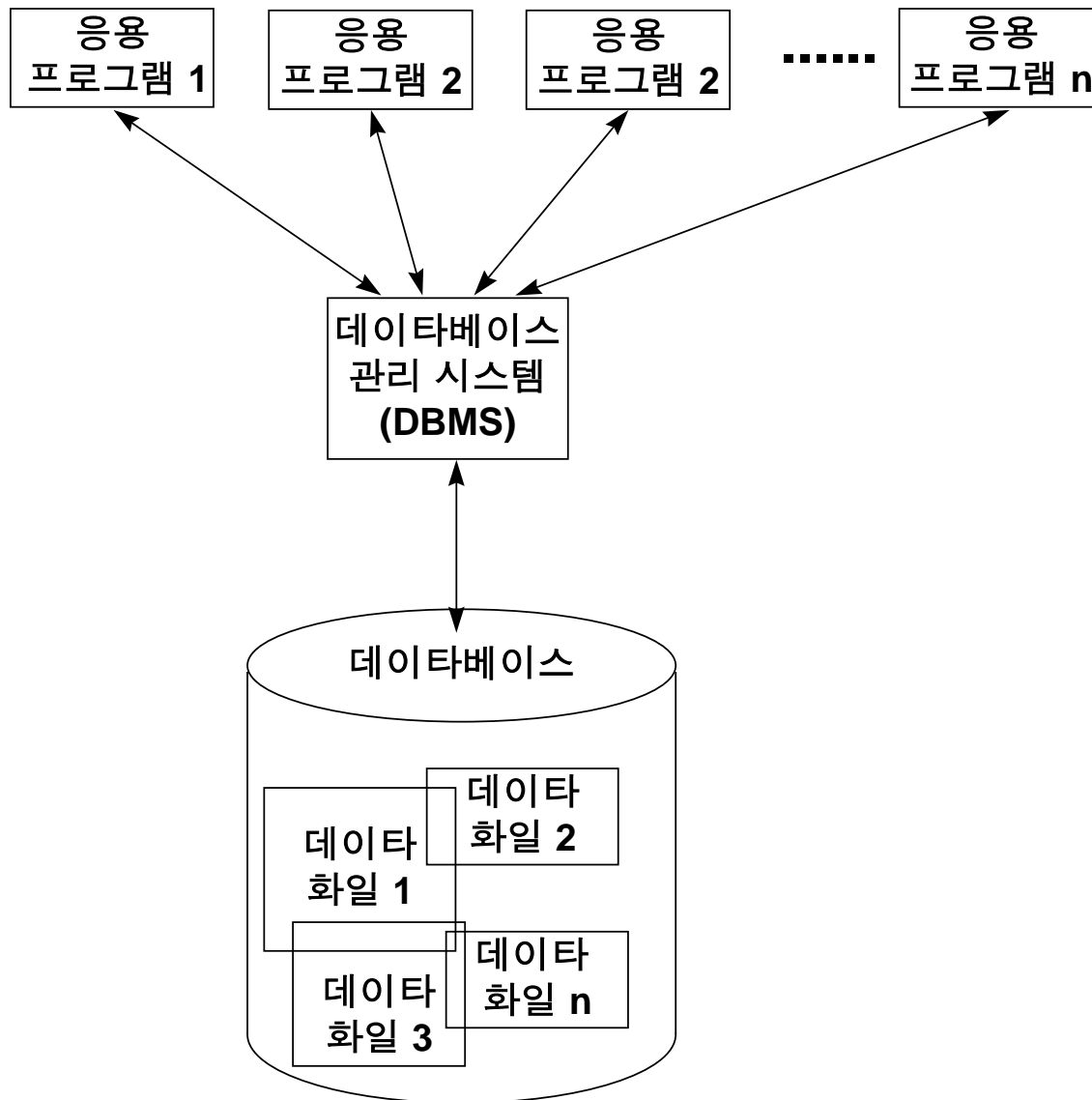
- 응용 프로그램과 데이터 간의 상호 의존관계
- 데이터의 구성방법이나 접근 방법의 변경시
- 관련 응용 프로그램도 같이 변경

❖ 데이터베이스 관리 시스템

◆ 데이터베이스 관리 시스템(DBMS)

- 응용 프로그램과 데이터의 중재자로서 모든 응용 프로그램들이 데이터베이스를 공유할 수 있게 관리해 주는 소프트웨어 시스템
- 기능
 - ◆ 데이터의 정의, 표현, 저장
 - ◆ 데이터 조작
 - ◆ 사용자 인터페이스 제공
 - ◆ 보안, 회복, 공유 제어, 무결성 기법 등을 포함, 데이터 제어

▶ 데이터베이스 관리 시스템과 데이터 파일



❖ 데이터베이스 관리 시스템의 장단점

1. 장점

- 데이터 중복(redundency)의 최소화
- 데이터 공유(sharing)
- 일관성(consistency) 유지
- 무결성(integrity) 유지
- 보안(security) 보장
- 표준화(standardization) 용이
- 상충되는 데이터 요구의 조정

2. 단점

- 운영비의 증대
- 데이터 처리 방법의 복잡성
- 어려운 백업(backup), 회복(recovery)
- 시스템의 취약성

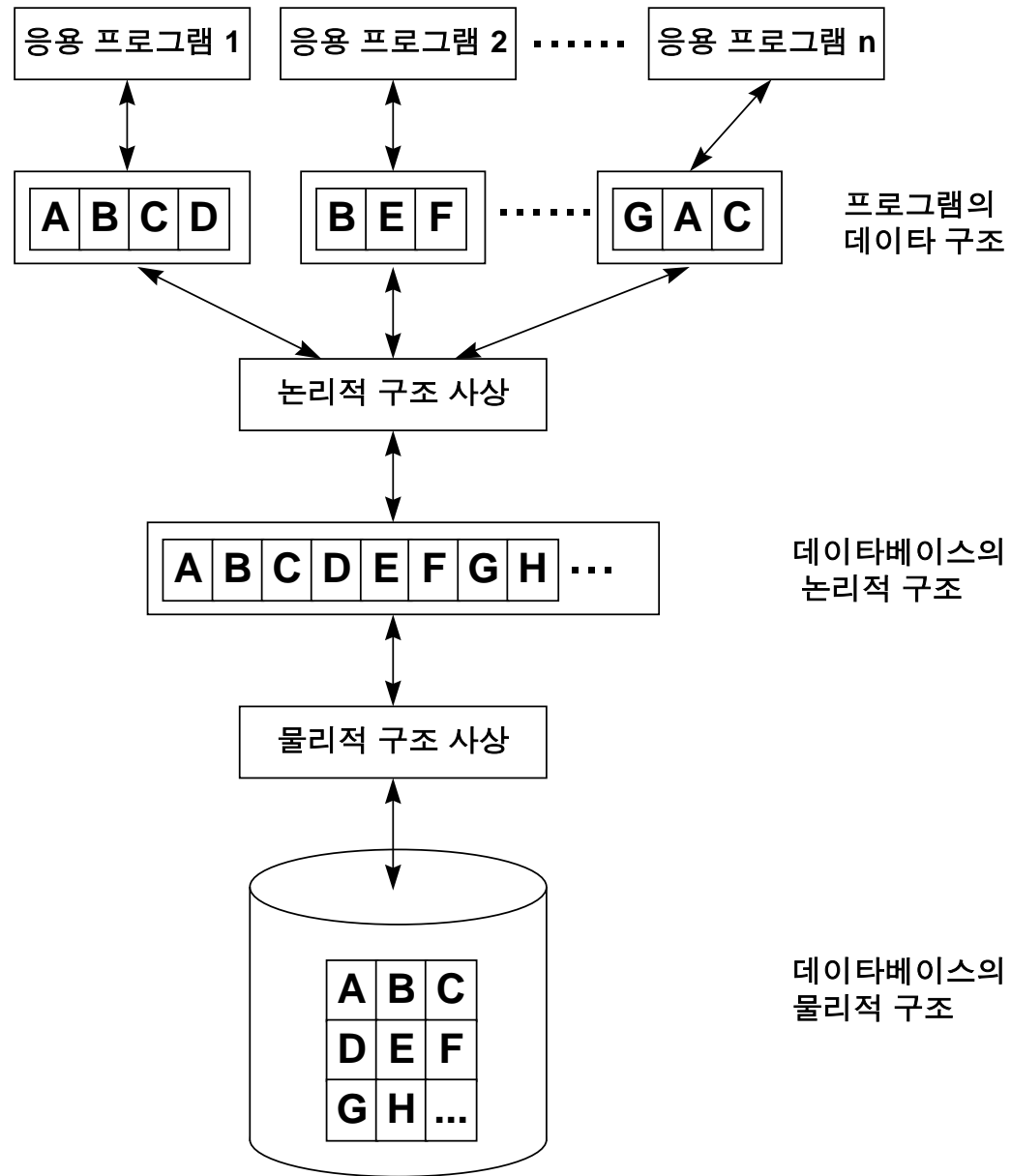
❖ 데이터 독립성

(1) 논리적 데이터 독립성

- 응용 프로그램에 영향을 주지 않고
논리적 데이터 구조의 변경 가능

(2) 물리적 데이터 독립성

- 응용 프로그램과 논리적 데이터 구조에 영향을 주지
않고 물리적 데이터 구조의 변경 가능



데이터 독립성과 데이터 구조간의 사상

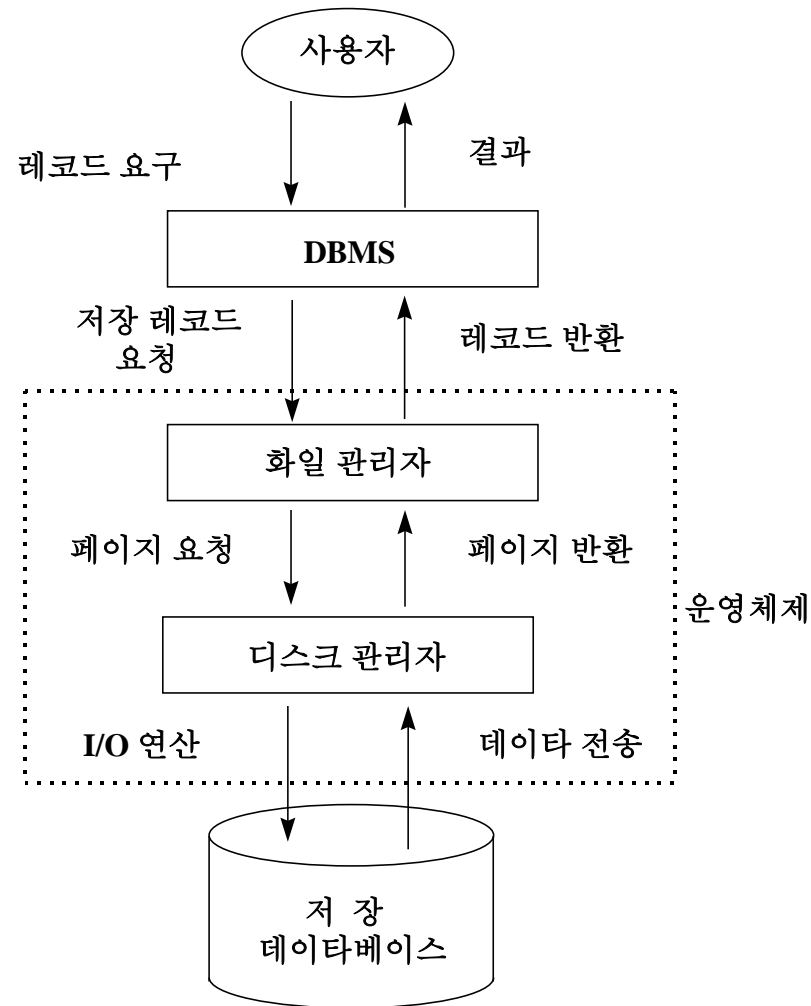
❖ 데이터베이스의 내부적 운영

1. 데이터베이스의 저장

- 데이터베이스의 내부적 운영
 - ◆ 데이터를 저장하는 방법과 접근에 관련된 작업
 - ◆ 디스크(DASD) 사용 - 디스크 접근 (디스크 I/O) 횟수를 최소화
- 저장구조
 - ◆ 디스크에 데이터가 배치, 저장 형식
 - ◆ 다수의 저장구조 지원
 - DB의 부분별로 적절한 저장
 - 성능요건 변경시 저장 구조 변경
 - ◆ 데이터베이스의 물리적 설계
 - DB의 사용 방법, 응용, 응용 실행빈도수에 따라 적절한 저장표현을 선정하는 과정

2. 데이터베이스의 접근

- 데이터베이스의 일반적인 접근 과정



(1) 디스크 관리자

- 기본 I/O 서비스 (basic I/O service)
 - ◆ 모든 물리적 I/O 연산에 대한 책임
 - ◆ 운영체제의 한 구성요소
- 물리적 디스크 주소
- 파일 관리자 지원
 - ◆ 디스크를 일정 크기의 페이지로 구성된 페이지 셀들의 논리적 집단으로 취급하도록 지원
 - ◆ 데이터 페이지 셀과 하나의 자유공간 페이지 셀
 - ◆ 페이지 셀 : 유일한 페이지 셀 ID를 갖는다.
- 디스크 관리
 - ◆ 페이지 번호 \leftarrow (사상) \rightarrow 물리적 디스크 주소
 - \rightarrow 파일 관리자를 장비에서 독립
 - ◆ 파일 관리자의 요청에 따라 페이지 셀에 대한 페이지의 할당과 회수

– 디스크 관리자의 페이지 관리 연산 – 화일 관리자가 명령

- ◆ 페이지 셀 S 로부터 페이지 P의 검색
 - ◆ 페이지 셀 S 내에서 페이지 P 의 교체
 - ◆ 페이지 셀 S 에 새로운 페이지 P 의 첨가
(자유공간 페이지 셀의 빈 페이지 할당)
 - ◆ 페이지 셀 S 에서 페이지 P 의 제거
(자유공간 페이지 셀에 반납)
- } 화일 관리자가 필요로 하는 연산

(2) 화일 관리자

- DBMS가 디스크를 저장 화일들의 집단으로 취급할 수 있도록 지원
- 저장화일(stored file)
 - ◆ 한 타입의 저장레코드 어커런스들의 집합
 - ◆ 한 페이지 셀은 하나 이상의 저장화일을 포함
 - ◆ 화일이름 또는 화일 ID로 식별
- 저장 레코드는 레코드번호 또는 레코드 ID(RID: Record Identifier)로 식별
 - ◆ 전체 디스크 내에서 유일
 - ◆ (페이지 번호, 페이지 오프셋)

– 화일 관리자의 화일 관리 연산

- ◆ 저장화일 f 에서 저장레코드 r 의 검색
- ◆ 저장화일 f 에 있는 저장레코드 r 의 대체
- ◆ 저장화일 f 에 새로운 레코드를 첨가하고 새로운 레코드 ID, r 을 부여
- ◆ 저장화일 f 에서 저장레코드 r 의 제거
- ◆ 새로운 저장화일 f 의 생성
- ◆ 저장화일 f 의 제거

3. 페이지 세트와 화일

- 디스크 관리자
 - ◆ 화일관리자가 물리적 디스크 I/O가 아닌
논리적인 페이지 I/O로 관리할 수 있게끔 지원
 - ◆ 페이지 관리(page management)

▶ 대학 데이터베이스

- ◆ 예)
 - 저장파일들은 28개의 페이지로 구성된 페이지 셀에 저장
 - 각 레코드들은 하나의 페이지를 차지

| 학생 | 학번 | 이름 | 학년 | 학과 |
|-----|-----|------|----|-----|
| S1: | 100 | 나 연묵 | 4 | 컴퓨터 |
| S2: | 200 | 이 찬영 | 3 | 전 기 |
| S3: | 300 | 정 기태 | 1 | 컴퓨터 |
| S4: | 400 | 송 병호 | 4 | 컴퓨터 |
| S5: | 500 | 박 종화 | 2 | 산 공 |

▶ 대학 데이터베이스

| 과목 | 과목 번호 | 과목 이름 | 학점 | 담당 교수 |
|-----|-------|--------|----|-------|
| C1: | C123 | 프로그래밍 | 3 | 김 성기 |
| C2: | C312 | 자료 구조 | 3 | 황 수찬 |
| C3: | C324 | 화일 처리 | 3 | 이 규철 |
| C4: | C413 | 데이터베이스 | 3 | 이 석호 |
| C5: | E412 | 반도체 | 3 | 홍 봉희 |

▶ 대학 데이터베이스

등록

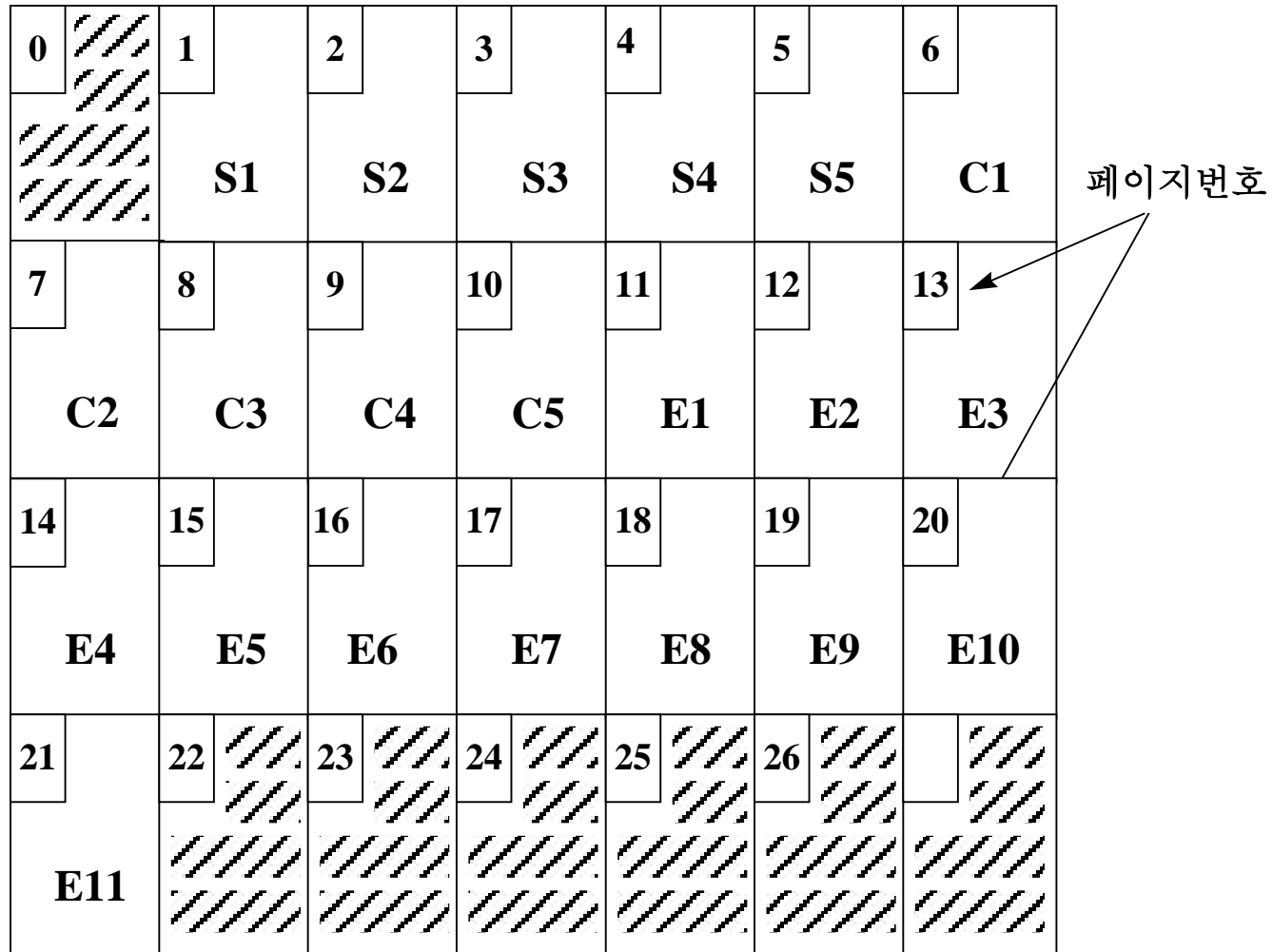
| | 학번 | 과목번호 | 성적 |
|------|-----|------|----|
| E1: | 100 | C413 | A |
| E2: | 100 | E412 | A |
| E3: | 200 | C123 | B |
| E4: | 300 | C312 | A |
| E5: | 300 | C324 | C |
| E6: | 300 | C413 | A |
| E7: | 400 | C312 | A |
| E8: | 400 | C324 | A |
| E9: | 400 | C413 | B |
| E10: | 400 | E412 | C |
| E11: | 500 | C312 | B |

▶ 연산















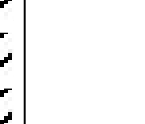



- ◆ 처음(빈 디스크) :
 - 하나의 자유 공간 페이지 셀만 존재(1~27)
 - 페이지 0 제외 : 디스크 디렉토리
- ◆ 화일관리자 : 학생 화일에 있는 5개의 레코드 적재
 - 디스크관리자 : 자유공간 페이지 셀의 페이지 1에서 5까지를 "학생 페이지 셀" 이라고 이름을 붙이고 할당
- ◆ 과목과 등록 화일에 대한 페이지 셀을 할당
 - 4개의 페이지 셀이 만들어짐
 - "학생"(1~5), "과목"(6~10), "등록"(11~21), "자유공간" 페이지 셀 (페이지 22~27)
- ◆ 화일관리자 : 새로운 학생 s6(학번 600)을 삽입
 - 디스크 관리자 : 첫번째 자유 페이지 (페이지 22)를 자유공간 페이지 셀에서 찾아서 학생 페이지 셀에 첨가

- ◆ 화일 관리자 : 새로운 과목 C6(E 515)를 삽입
 - 디스크 관리자 : 자유공간 페이지 셸에서
첫번째 자유페이지 (페이지 2)를 찾아서
과목 페이지 셸에 첨가
- ◆ 화일 관리자 : S4를 삭제
 - 디스크 관리자 : S4가 저장되어 있던 페이지
(페이지 4)를 자유공간 페이지 셸에 반납

▶ 대학 데이터베이스의 초기 적재 후의 디스크 배치도

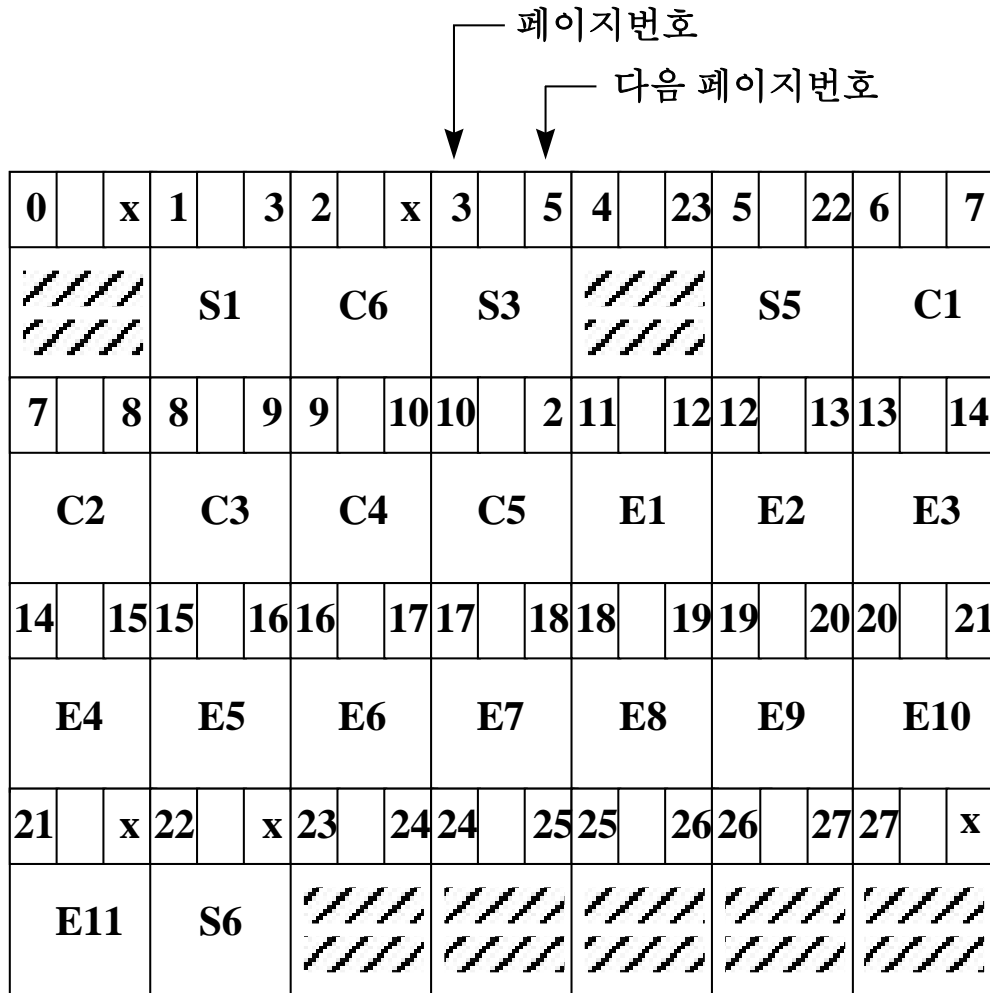


▶ 일련의 삭제 연산이 수행된 뒤의 디스크 배치도

| | | | | | | | | | | | | | |
|---|---|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 |  | 1 | | 2 | | 3 | | 4 |  | 5 | | 6 | |
|  | S1 | | C6 | | S3 | |  | S5 | | C1 | | | |
| 7 | | 8 | | 9 | | 10 | | 11 | | 12 | | 13 | |
| C2 | | C3 | | C4 | | C5 | | E1 | | E2 | | E3 | |
| 14 | | 15 | | 16 | | 17 | | 18 | | 19 | | 20 | |
| E4 | | E5 | | E6 | | E7 | | E8 | | E9 | | E10 | |
| 21 | | 22 | | 23 |  | 24 |  | 25 |  | 26 |  | 27 |  |
| E11 | | S6 | |  |  |  |  |  |  |  |  |  | |

- 삽입, 삭제 연산 실행후에는 페이지들의 물리적 인접성이 없어짐
- 포인터 표현 방법
 - ◆ 한 페이지 셀에서 페이지의 논리적 순서가 물리적 인접으로 표현되지 않음
 - 페이지 : 페이지 헤드 - 제어정보 저장
 - 포인터 : 논리적 순서에 따른 다음 페이지의 물리적 주소
 - 다음 페이지 포인터는 디스크 관리자가 관리 (화일 관리자는 무관)

▶ 페이지 헤드에 “다음 페이지” 포인터를 포함시킨 경우의 디스크 배치도



▶ 디스크 디렉토리(페이지 셀 디렉토리)

실린더 0, 트랙 0에 위치

- 디스크에 있는 모든 페이지 셀의 리스트와
각 페이지 셀의 첫번째 페이지에 대한 포인터 저장

| | | | | | | | | | | | | |
|---|-----|---|------|-----|------|---|-----|---|-----|---|-----|----|
| 0 | | × | | | | | | | | | | |
| <table><tr><td>페이지셀</td><td>주 소</td></tr><tr><td>자유공간</td><td>4</td></tr><tr><td>학 생</td><td>1</td></tr><tr><td>과 목</td><td>6</td></tr><tr><td>등 록</td><td>11</td></tr></table> | | | 페이지셀 | 주 소 | 자유공간 | 4 | 학 생 | 1 | 과 목 | 6 | 등 록 | 11 |
| 페이지셀 | 주 소 | | | | | | | | | | | |
| 자유공간 | 4 | | | | | | | | | | | |
| 학 생 | 1 | | | | | | | | | | | |
| 과 목 | 6 | | | | | | | | | | | |
| 등 록 | 11 | | | | | | | | | | | |





디스크 디렉토리

▶ 화일 관리자

- 저장 레코드 관리 (stored record management)
 - ◆ DBMS가 페이지 I/O 에 대한 세부적인 사항에 대해 알 필요없이 저장화일과 저장 레코드만으로 동작하게 함
- 예
 - ◆ 하나의 페이지에 여러개의 레코드 저장
 - ◆ 학생 레코드에 대한 논리적 순서는 학번순

▶ 5개의 학생 레코드를 처음 적재한 페이지 P의 배치도

- ① 페이지 p에 5개의 학생레코드(S1 ~ S5)가
삽입되어 있다고 가정

| | | | |
|---|-----------|---|-----|
| P | | | ● → |
| S1 | S2 | S3 | |
| S4 | S5 |  | |
| | |  | |
|  | | | |
|  | | | |

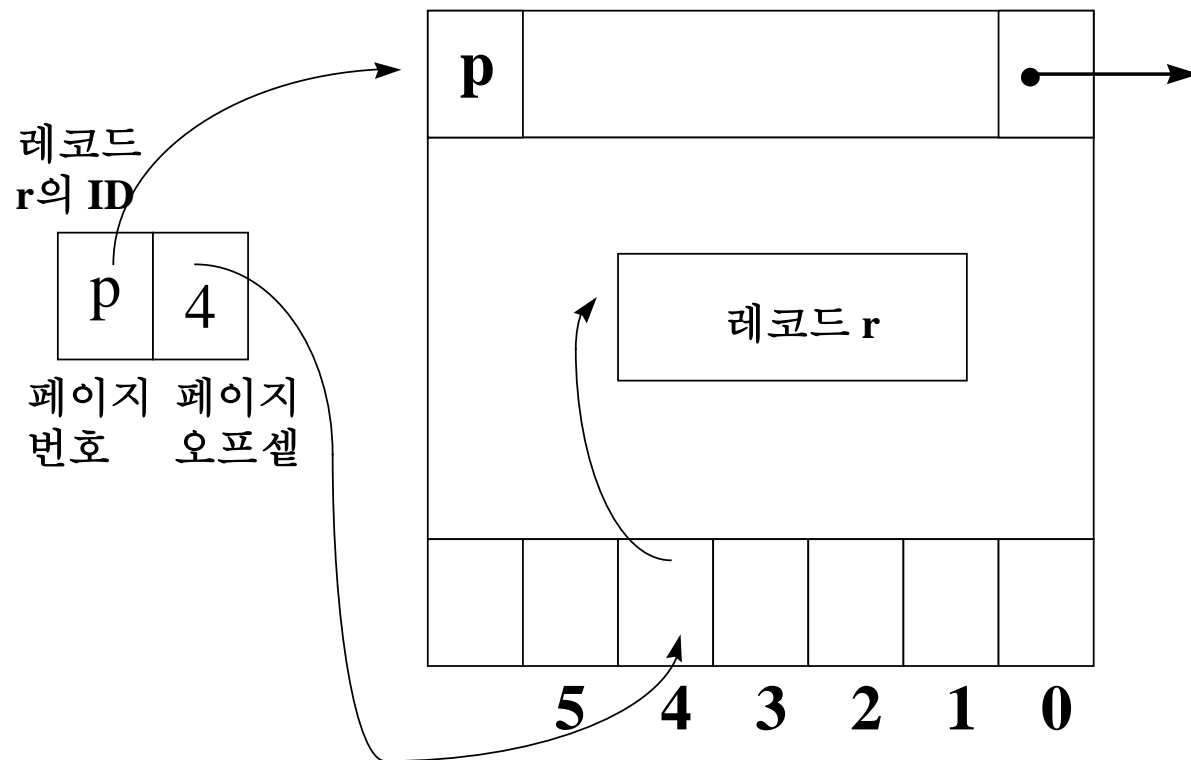
- ② DBMS : 학생 레코드 S9(학번 900)의 삽입 요청
페이지 p의 학생레코드 S5 바로 다음에 저장
- ③ DBMS : 레코드 S2의 삭제 요청
 - 페이지 p에 있는 학생 레코드 S2를 삭제하고
뒤에 있는 레코드들을 모두 앞으로 당김
- ④ DBMS : 레코드 S7(학번 700)의 삽입 요청
 - 학생레코드 S5 다음에 들어가야 되므로 학생
레코드 S9를 뒤로 옮김

▶ S2가 삭제되고 S9와 S70이 삽입된 뒤의 페이지 P의 배치도

| | | | |
|----------------------|----|----|-----|
| P | | | • → |
| S1 | S2 | S3 | |
| S5 | S7 | S9 | |
| //////////////////// | | | |
| | | | |

- 한 페이지 내에서 저장레코드의 논리적 순서는
그 페이지내에서의 물리적 순서로 표현 가능
- 레코드들이 페이지 내에서 이동
- 레코드들을 모두 페이지 윗쪽으로 저장
 - 아래쪽은 계속적으로 자유공간으로 유지

▶ RID의 구현



- RID = (페이지 번호 p, 오프셋)
- 페이지 오프셋 = 페이지내에서의 레코드 위치
- 레코드가 한 페이지내에서 이동할때마다 RID의 변경없이 페이지 오프셋의 내용(포인터)만 변경
- 최악의 경우 두번째 접근으로 원하는 레코드 검색가능
 - 두번 접근 : 해당 페이지가 오버플로우가 되어 다른 페이지로 저장된 경우