Chapter 9

전체 텍스트 검색과 파티션

Contents

CHAPTER 9 전체 텍스트 검색과 파티션

MySQL의 성능을 향상하기 위한 방법으로 전체 텍스트 검색과 파티션에 대해 알아본다.

SECTION 01 전체 텍스트 검색

전체 텍스트 검색 개요

- 전체 텍스트 검색
  - 긴 문자로 구성된 구조화 되지 않은 텍스트 데이터(예로, 신문 기사) 등을 빠르게 검색하기 위한 부가적인 MySQL의 기능
  - 저장된 텍스트의 키워드 기반의 쿼리를 위해서 빠른 인덱싱 가능

[그림 11-1] 신문기사 테이블과 신문기사 인덱스 개념도

SECTION 01 전체 텍스트 검색

전체 텍스트 검색 개요

- 전체 텍스트 검색
  - 신문 기사 검색

```
SELECT * FROM 신문기사_테이블 WHERE 신문기사내용 = '교통 사고의 증가로 인해 오늘 ~~~~';
```

  - 교통'을 키워드로 가진 기사 검색

```
SELECT * FROM 신문기사_테이블 WHERE 신문기사내용 LIKE '교통%'
```

  - 키워드가 중간에 들어간 경우 인덱스 사용 불가, 서버에 엄청난 부하 발생

```
SELECT * FROM 신문기사_테이블 WHERE 신문기사내용 LIKE '%교통%'
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

- 전체 텍스트 인덱스(FULLTEXT Index) 생성
  - 텍스트로 이루어진 문자열 데이터의 내용으로 생성한 인덱스
  - 텍스트 인덱스 생성 형식

```
형식1 :
CREATE TABLE 테이블이름 (
  ...
  열 이름 데이터형식,
  ... ,
  FULLTEXT 인덱스이름 (열 이름)
);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 인덱스(FULLTEXT Index) 생성

- 텍스트 인덱스 생성 형식

형식2 :

```
CREATE TABLE 테이블이름 (
  ...
  열 이름 데이터형식,
  ...
);
ALTER TABLE 테이블이름
   ADD FULLTEXT (열 이름) ;
```

형식3 :

```
CREATE TABLE 테이블이름 (
  ...
  열 이름 데이터형식,
  ...
);
CREATE FULLTEXT INDEX 인덱스이름
ON 테이블이름 (열 이름);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 인덱스(FULLTEXT Index) 생성

- 전체 텍스트 인덱스의 특징
  - InnoDB와 MyISAM 테이블만 지원
  - char, varchar, text의 열에만 생성 가능
  - 인덱스 힌트 사용 일부 제한
  - 여러 개 열에 FULLTEXT 인덱스를 지정 가능

◦ 전체 텍스트 인덱스 삭제

```
형식 :
ALTER TABLE 테이블이름
    DROP INDEX FULLTEXT(열 이름) ;
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 중지 단어

- 전체 텍스트 인덱스는 긴 문장에 대해서 인덱스 생성하기 때문에 양이 방대함
  - 검색시 무시할 만한 단어들은 전체 텍스트 인덱스로 생성하지 않음
- MySQL 8.0은 INFORMATION_SCHEMA.INNODB_FT_DEFAULT_STOPWORD 테이블 존재
  - 36개의 중지 단어 가지고 있음

value
a
about
an
are
as
at
be
by

[그림 11-2] MySQL이 가지고 있는 중지 단어

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 검색을 위한 쿼리

- 전체 텍스트 인덱스를 생성한 후 전체 텍스트 인덱스 이용하기 위한 쿼리
  - 일반 SELECT문의 WHERE절에 MATCH( ) AGAINST( ) 사용

```
형식 :
MATCH (col1,col2,...) AGAINST (expr [search_modifier])

search_modifier:
  {
        IN NATURAL LANGUAGE MODE
      | IN NATURAL LANGUAGE MODE WITH QUERY EXPANSION
      | IN BOOLEAN MODE
      | WITH QUERY EXPANSION
  }
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 검색을 위한 쿼리

- 자연어 검색
  - 정확한 단어를 검색
  - 특별히 옵션 지정하지 않거나 IN NATURAL LANGUAGE MODE 사용
  - '영화'라는 단어가 들어간 기사를 찾을 때 사용

```
SELECT * FROM newspaper
  WHERE MATCH(article) AGAINST('영화');
```

  - '영화' 또는 '배우' en 단어 중 하나가 포함된 기사를 찾을 때 사용

```
SELECT * FROM newspaper
  WHERE MATCH(article) AGAINST('영화 배우');
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 검색을 위한 쿼리

- 불린 모드 검색
  - 단어나 문장이 정확히 **일치하지 않는 것도 검색**하는 것
  - IN BOOLEAN MODE 옵션 필요
  - 필수 '+'
  - 제외 '-'
  - 부분 검색 '*' 등 연산자 등의 다양한 연산자 지원
  - '영화를', '영화가', '영화는' 등의 '영화'가 앞에 들어간 결과를 검색하고 싶을 때 사용

```
SELECT * FROM newspaper
   WHERE MATCH(article) AGAINST('영화*' IN BOOLEAN MODE);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스

◦ 전체 텍스트 검색을 위한 쿼리

- 불린 모드 검색

  • '영화 배우' 단어가 정확히 들어 있는 기사의 내용을 검색하고 싶을 때 사용

```
SELECT * FROM newspaper
   WHERE MATCH(article) AGAINST('영화 배우' IN BOOLEAN MODE);
```

  • '영화 배우' 단어가 들어 있는 기사 중 '공포'의 내용이 꼭 들어간 결과만 검색하고 싶을 때 사용

```
SELECT * FROM newspaper
   WHERE MATCH(article) AGAINST('영화 배우 +공포' IN BOOLEAN MODE);
```

  • '영화 배우' 단어가 들어 있는 기사 중에서 '남자'의 내용은 검색에서 제외하고 싶을 때 사용

```
SELECT * FROM newspaper
   WHERE MATCH(article) AGAINST('영화 배우 -남자' IN BOOLEAN MODE);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- MySQL 환경 설정 파일 my.ini 파일의 [mysqld]에 innodb_ft_min_token_size = 2 추가
  - '남성'등의 2글자 단어를 검색위해 2로 변경(3으로 설정되어 있음)
- MySQL 서비스 재시작 또는 컴퓨터 재부팅
- 데이터베이스 및 테이블 생성

```
CREATE DATABASE IF NOT EXISTS FulltextDB;
USE FulltextDB;
DROP TABLE IF EXISTS FulltextTbl;
CREATE TABLE FulltextTbl (
   id int AUTO_INCREMENT PRIMARY KEY,          -- 고유 번호
   title VARCHAR(15) NOT NULL,                 -- 영화 제목
   description VARCHAR(1000)                   -- 영화 내용 요약
);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- 데이터 입력 및 '남자' 단어 검색

```
INSERT INTO FulltextTbl VALUES
(NULL, '광해, 왕이 된 남자','왕위를 둘러싼 권력 다툼과 당쟁으로 혼란이 극에 달한 광해군 8년'),
(NULL, '간첩','남한 내에 고장간첩 5만 명이 암약하고 있으며 특히 권력 핵심부에도 침투해있다.'),
(NULL, '남자가 사랑할 때', '대책 없는 한 남자이야기. 형 집에 얹혀 살며 조카한테 무시당하는 남자'),
(NULL, '레지던트 이블 5','인류 구원의 마지막 퍼즐, 이 여자가 모든 것을 끝낸다.'),
(NULL, '파괴자들','사랑은 모든 것을 파괴한다! 한 여자를 구하기 위한, 두 남자의 잔인한 액션 본능!');
```

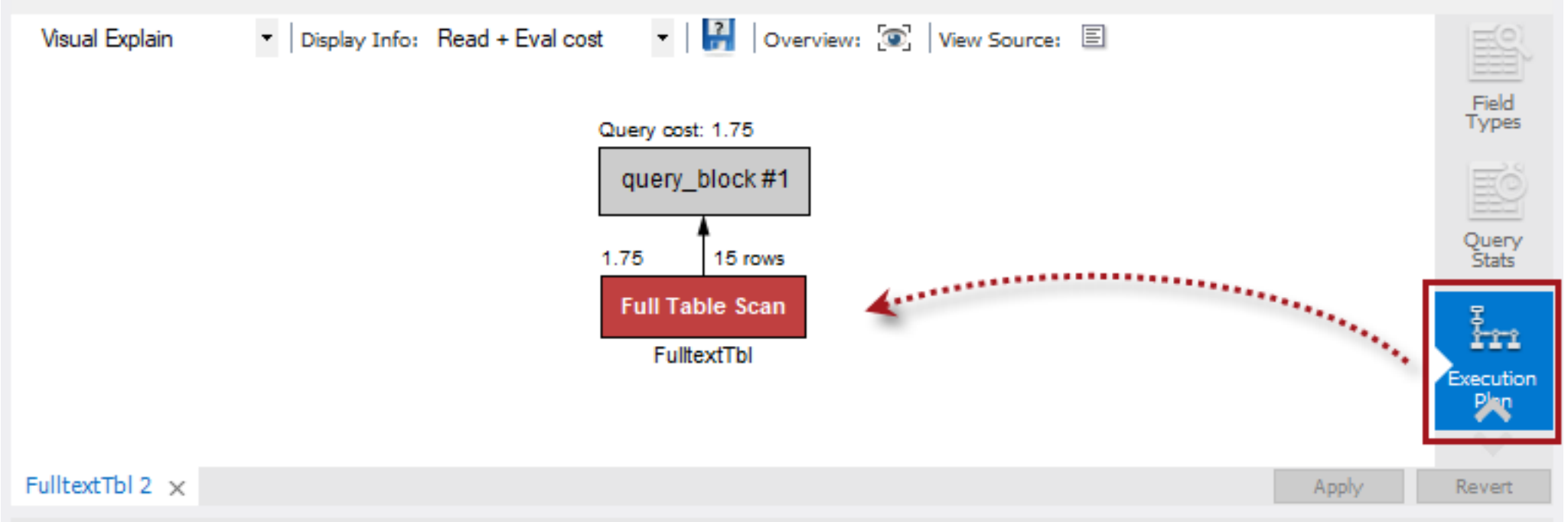
```
SELECT * FROM FulltextTbl WHERE description LIKE '%남자%' ;
```

	id	title	description
▶	3	남자가 사랑할 때	대책 없는 한 남자이야기. 형 집에 얹혀 살며 조카한테 무시당하는 남자
	5	파괴자들	사랑은 모든 것을 파괴한다! 한 여자를 구하기 위한, 두 남자의 잔인한 액션 본능!

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

- 전체 텍스트 검색을 위한 쿼리
  - 전체 테이블 검색 실행 계획 확인



SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- 전체 텍스트 인덱스 생성

```
CREATE FULLTEXT INDEX idx_description ON FulltextTbl(description);
```

- 정보 확인

```
SHOW INDEX FROM FulltextTbl;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
fulltexttbl	0	PRIMARY	1	id	A	15	NULL	NULL		BTREE		
fulltexttbl	1	idx_description	1	description	NULL	15	NULL	NULL	YES	FULLTEXT		

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

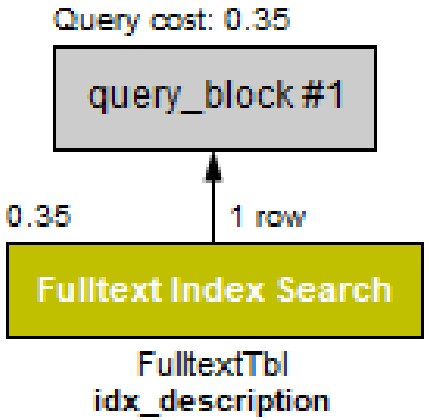
◦ 전체 텍스트 검색을 위한 쿼리

- 전체 텍스트 인덱스 활용한 검색

```
SELECT * FROM FulltextTbl WHERE MATCH(description) AGAINST('남자*' IN BOOLEAN MODE);
```

	id	title	description
▶	3	남자가 사랑할 때	대책 없는 한 남자이야기. 형 집에 얹혀 살며 조카한테 무시당하는 남자
	5	파괴자들	사랑은 모든 것을 파괴한다! 한 여자를 구하기 위한, 두 남자의 잔인한 액션 본능!

- 전체 텍스트 인덱스 검색 실행 계획



SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- '남자' 또는 '여자'가 들어간 행 검색, 매치되는 점수 출력

```
SELECT *, MATCH(description) AGAINST('남자* 여자*' IN BOOLEAN MODE) AS 점수
    FROM FulltextTbl WHERE MATCH(description) AGAINST('남자* 여자*' IN BOOLEAN MODE);
```

id	title	description	점수
5	파괴자들	사랑은 모든 것을 파괴한다! 한 여자를 구하기 위한, 두 남자의 잔인한 액션 본능!	0.9771181344985962
3	남자가 사랑할 때	대책 없는 한 남자 이야기. 형 집에 얹혀 살며 조카한테 무시당하는 남자	0.4885590672492981
4	레지던트 이블 5	인류 구원의 마지막 퍼즐, 이 여자가 모든 것을 끝낸다.	0.4885590672492981
9	8월의 크리스마스	시한부 인생 사진사와 여자 주차 단속원과의 미묘한 사랑	0.4885590672492981

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- '남자' 와 '여자'가 둘다 들어 있는 영화 검색 : + 연산자 사용

```
SELECT * FROM FulltextTbl
    WHERE MATCH(description) AGAINST('+남자* +여자*' IN BOOLEAN MODE);
```

- '남자'가 들어 있는 영화 중에서 '여자'가 들어있는 영화 제외 : - 연산자 사용

```
SELECT * FROM FulltextTbl
    WHERE MATCH(description) AGAINST('남자* -여자*' IN BOOLEAN MODE);
```

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 전체 텍스트 검색을 위한 쿼리

- 전체 텍스트 인덱스로 만들어진 단어 확인

```
SET GLOBAL innodb_ft_aux_table = 'fulltextdb/fulltexttbl'; -- 모두 소문자
SELECT word, doc_count, doc_id, position
    FROM INFORMATION_SCHEMA.INNODB_FT_INDEX_TABLE;
```

word	doc_count	doc_id	position
같은	1	8	61
[illegible]	[illegible]	[illegible]	7
구하기	1	7	52
국가대표팀이	1	13	63
권력	2	3	20
권력	2	4	69
귀도는	1	15	0
그는	1	14	0
그리고	1	14	58
극에	1	3	60
급조된다	1	13	82
기적	1	8	[illegible]

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 중지 단어 추가

- 앞에서 생성한 전체 텍스트 인덱스 삭제

```
DROP INDEX idx_description ON FulltextTbl;
```

- 중지 단어 저장할 테이블 생성

```
CREATE TABLE user_stopword (value VARCHAR(30));
```

- 중지 단어 입력 : '그는', '그리고', '극에'

```
INSERT INTO user_stopword VALUES ('그는'), ('그리고'), ('극에');
```

- 중지 단어용 테이블 시스템 변수 설정 : **DB 이름과 테이블 이름은 모두 소문자**

```
SET GLOBAL innodb_ft_server_stopword_table = 'fulltextdb/user_stopword';
SHOW GLOBAL VARIABLES LIKE 'innodb_ft_server_stopword_table';
```

	Variable_name	Value
▶	innodb_ft_server_stopword_table	fulltextdb/user_stopword

SECTION 01 전체 텍스트 검색

전체 텍스트 인덱스 실습

◦ 중지 단어 추가

- 전체 텍스트 인덱스 생성

```
CREATE FULLTEXT INDEX idx_description ON FulltextTbl(description);
```

- 전체 텍스트 인덱스에 생성된 단어 확인

```
SELECT word, doc_count, doc_id, position
  FROM INFORMATION_SCHEMA.INNODB_FT_INDEX_TABLE;
```

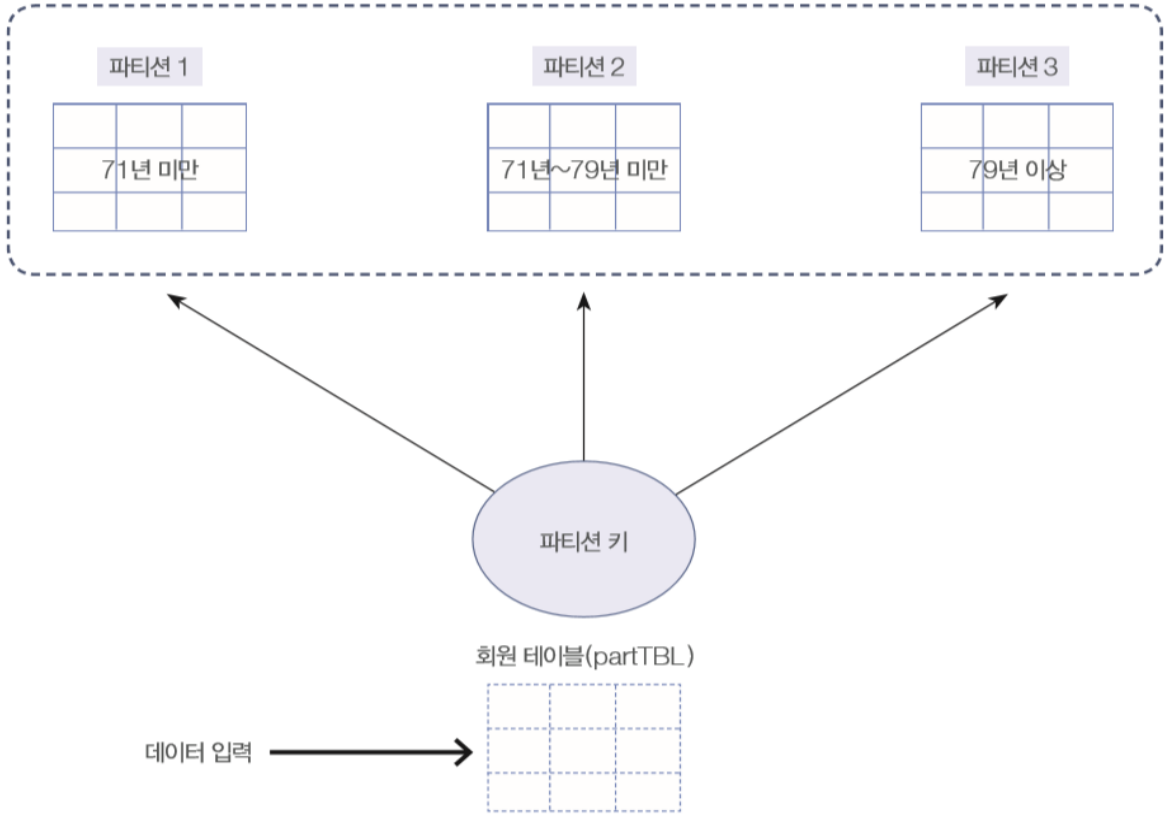
- 중지 단어 설정한 단어가 보이지 않음
  - '그는', '그리고', '극에'

word	doc_count	doc_id	position
권력	2	4	69
귀도는	1	15	0
급조된다	1	13	82
기적	1	8	54
끝낸다	1	6	63
남자	1	5	89
남자의	1	7	

SECTION 02 파티션

파티션 개요와 실습

- 파티션
  - 대량의 테이블을 물리적으로 여러 개의 테이블로 쪼개기
  - 데이터의 분포 특성, 자주 사용되는 쿼리문이 무엇인지에 따라서 효율에 차이 있음



[그림 11-12] 파티션 개념도

SECTION 02 파티션

파티션 개요와 실습

◦ 파티션 구성

파티션 이름	part1	part2	part3
파티션 일련번호	1	2	3
값	값 < 1971	1971 <= 값 < 1979	1979 <= 값 < 최대

```
CREATE DATABASE IF NOT EXISTS partDB;
USE partDB;
DROP TABLE IF EXISTS partTBL;
CREATE TABLE partTBL (
  userID  CHAR(8) NOT NULL, -- Primary Key로 지정하면 안 됨
  name  VARCHAR(10) NOT NULL,
  birthYear INT  NOT NULL,
  addr CHAR(2) NOT NULL )
PARTITION BY RANGE(birthYear) (
    PARTITION part1 VALUES LESS THAN (1971),
    PARTITION part2 VALUES LESS THAN (1979),
    PARTITION part3 VALUES LESS THAN MAXVALUE
);
```

SECTION 02 파티션

파티션 개요와 실습

◦ 파티션 구성

- 파티션 테이블에는 Primary Key 지정 하면 안됨
- 데이터 입력 : 입력됨과 동시에 파티션 키에 의해서 데이터가 각 파티션으로 나뉘어짐

```
INSERT INTO partTBL
    SELECT userID, name, birthYear, addr FROM sqlDB.userTbl;
SELECT * FROM partTBL;
```

userID	name	birthYear	addr
JKW	조관우	1965	경기
JYP	조용필	1950	경기
LJB	임재범	1963	서울
YJS	윤종신	1969	경남
BBK	바비킴	1973	서울
EJW	은지원	1972	경북
KKH	김경호	1971	전남
KBS	김범수	1979	경남
LSG	이승기	1987	서울
SSK	성시경	1979	서울

파티션1 (1971 미만)

파티션2 (1971~1979 미만)

파티션3 (1979 이상)

SECTION 02 파티션

파티션 개요와 실습

◦ 파티션 구성

- 파티션 확인

• INFORMATION_SCHEMA 데이터베이스의 PARTITIONS 테이블에 관련 정보 있음

```
SELECT TABLE_SCHEMA, TABLE_NAME, PARTITION_NAME, PARTITION_ORDINAL_POSITION, TABLE_
ROWS
    FROM INFORMATION_SCHEMA.PARTITIONS
    WHERE TABLE_NAME =  'parttbl';
```

TABLE_SCHEMA	TABLE_NAME	PARTITION_NAME	PARTITION_ORDINAL_POSITION	TABLE_ROWS
partdb	parttbl	part1	1	4
partdb	parttbl	part2	2	3
partdb	parttbl	part3	3	3

SECTION 02 파티션

파티션 개요와 실습

◦ 파티션 구성

- 1965년 이전 출생한 회원 조회

```
SELECT * FROM partTBL WHERE birthYear <= 1965 ;
```

  - 결과는 3명 출력

- 어느 파티션을 사용했는지 확인하려면 쿼리문 앞에 EXPLAIN문을 붙임

```
EXPLAIN  SELECT * FROM partTBL WHERE birthYear <= 1965;
```

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	partTBL	part1	ALL	NULL	NULL	NULL	NULL	4	33.33	Using where

파티션 개요와 실습

◦ 파티션 나누기

- 파티션3을 1979 ~ 1986미만(파티션3)과 1986 이상(파티션4)로 분리
  - 파티션 분리 : ALTER TABLE... REORGANIZE PARTITION문 사용
  - 파티션을 재구성 : OPTIMIZE TABLE문 사용

```
ALTER TABLE partTBL
    REORGANIZE PARTITION part3 INTO (
            PARTITION part3 VALUES LESS THAN (1986),
            PARTITION part4 VALUES LESS THAN MAXVALUE
    );
OPTIMIZE TABLE partTBL;
```

TABLE_SCHEMA	TABLE_NAME	PARTITION_NAME	PARTITION_ORDINAL_POSITION	TABLE_ROWS
partdb	parttbl	part1	1	4
partdb	parttbl	part2	2	3
partdb	parttbl	part3	3	2
partdb	parttbl	part4	4	1

파티션 개요와 실습

- 파티션 합치기
  - 파티션1과 파티션2를 합쳐서 파티션12로 합치기

```
ALTER TABLE partTBL
   REORGANIZE PARTITION part1, part2 INTO (
            PARTITION part12 VALUES LESS THAN (1979)
   );
OPTIMIZE TABLE partTBL;
```

TABLE_SCHEMA	TABLE_NAME	PARTITION_NAME	PARTITION_ORDINAL_POSITION	TABLE_ROWS
partdb	parttbl	part12	1	7
partdb	parttbl	part3	2	2
partdb	parttbl	part4	3	1

SECTION 02 파티션

파티션 개요와 실습

◦ 파티션 삭제

- 파티션12 삭제

```
ALTER TABLE partTBL DROP PARTITION part12;
OPTIMIZE TABLE partTBL;
```

- 데이터 조회

```
SELECT * FROM partTBL;
```

userID	name	birthYear	addr
KBS	김범수	1979	경남
SSK	성시경	1979	서울
LSG	이승기	1987	서울

SECTION 02 파티션

파티션 개요와 실습

- 파티션의 특징
  - 파티션 테이블에 외래 키 설정 불가
    - 단독으로 사용되는 테이블에만 파티션 설정
  - 스토어드 프로시저, 스토어드 함수, 사용자 변수 등은 파티션 함수나 식에 사용 불가
  - 임시 테이블은 파티션 기능을 사용 불가
  - 파티션 키에는 일부 함수만 사용 가능
  - 파티션 개수는 최대 8,192개까지 지원
  - 레인지 파티션은 숫자형의 연속된 범위 사용
  - 리스트 파티션
    - 숫자형 또는 문자형의 연속되지 않은 파티션 키 값 지정
    - MAXVALUE 사용 불가, 모든 경우의 파티션 키 값 지정