5. 기본 SQL 작성하기-DML

데이터조작어(DML:DATA MANIPULATION LANGUAGE)

- ✓ DML은 테이블에서 데이터를 삽입, 조회, 삭제, 수정하기 위해 사용하는 질의어이다.
- ✓ INSERT, SELECT, DELETE, UPDATE

<< 참고 >>

SQL 명령어는 3가지 종류로 나눈다.

DDL(Definition): CREATE, DROP, ALTER 등

DML(Manipulation) : SELECT, INSERT, DELETE, UPDATE

DCL(Contol): GRANT, REVOKE

데이터 삽입: INSERT INTO ~

- ◆ 테이블에 행(튜플)을 삽입(INSERT문)
 - 한 행을 삽입

```
INSERT INTO 테이블명(컬럼1, 컬럼2, 컬럼3, ...)
VALUES (값1, 값 2, 값 3, ...);
```

INSERT INTO nation(n_code, n_name, number) **VALUES** (1, 'korea', 50);

nation 테이블 n_code, n_name, number 컬럼에 각각 1, korea, 50의 값을 입력한다

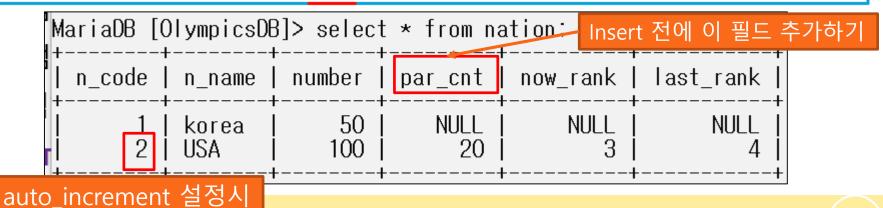
INSERT INTO ~

- ◆ 테이블에 행을 삽입(INSERT문)
 - 컬럼명 생략 가능

INSERT INTO 테이블명 VALUES (값1, 값 2, 값 3, ...);

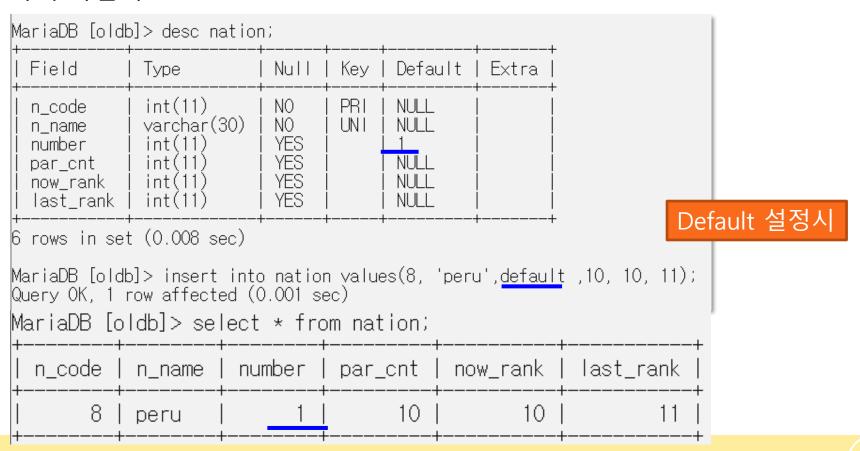
- 컬럼 리스트의 개수와 값 리스트의 **개수가 동일**할 때
- 컬럼리스트를 생략하면 모든 컬럼을 의미하므로 컬럼 개수 만큼 값을 입력해야 한다. 또한 값들은 **컬럼 순서와 동일**하게 입력해야 한다.

INSERT INTO nation VALUES (null, 'USA', 100, 20, 3, 4);



INSERT INTO ~

- ◆ 테이블에 행을 삽입(INSERT문)
 - 컬럼에 default 속성이 있을 경우 해당 컬럼에 default 라고 쓰면 default 값이 추가된다



AUTO_INCREMENT

◆ AUTO_INCREMENT 초기화

n_code 8번이 있는 상태에서 새로운 데이터 추가하면, 9번으로 생김.

ALTER TABLE 테이블명 AUTO_INCREMENT=[시작할 값];

이 경우 테이블에 새로 시작할 값보다 큰 값이 있으면 안된다.

ALTER TABLE nation AUTO_INCREMENT=10 ;

MariaDB [olympicsdb]> select * from nation;					
n_code	n_name	number	par_cnt	now_rank	last_rank
1 2 3 4 5 6	korea USA CHINA FRANCE JAPAN ENGLAND CHINA	50 100 150 90 80 95 150	NULL 20 25 20 20 19 25	NULL 3 1 2 10 5	NULL 4 5 3 2 1 5

INSERT INTO ~

- ◆ 테이블에 행을 삽입(INSERT문)
 - 여러 행을 삽입

INSERT INTO 테이블명[(컬럼1, 컬럼2, 컬럼3, ...)]

INSERT INTO nation VALUES (null, 'CHINA', 150, 25, 1, 5), (NULL, 'FRANCE', 90, 20, 2, 3), (NULL, 'JAPAN', 80, 20, 10, 2), (NULL, 'ENGLAND', 95, 19, 5, 1);

MariaDB [(OlympicsDB]> select	* from nat	tion;	
n_code	n_name	number	par_cnt	now_rank	last_rank
i 3 4 5	korea USA CHINA FRANCE JAPAN ENGLAND	50 100 150 90 80 95	NULL 20 25 20 20 19	NULL 2 1 2 2 10 5	NULL 4 5 3 2 1

AUTO_INCREMENT

◆ AUTO_INCREMENT 를 사용했어도 값을 직접 INSERT 가능하다.

INSERT INTO nation VALUES (11, 'USA', 100, 20, 3, 4);

MariaDB [olympicsdb]> select * from nation;					
n_code	n_name	number	par_cnt	now_rank	last_rank
1 2 3 4 5 6 10 11	korea USA CHINA FRANCE JAPAN ENGLAND CHINA USA	50 100 150 90 80 95 150 100	NULL 20 25 20 20 19 25 20	NULL 3 1 2 10 5 1	NULL 4 5 3 2 1 5 4

◆ 테이블에서 조건에 맞는 데이터를 검색(SELECT 문)

SELECT [DISTINCT] 컬럼.... FROM 테이블명
[WHERE 조건]
[GROUP BY 컬럼명]
[]: 생략가능
[HAVING 조건]
[ORDER BY 컬럼명 [ASC | DESC];

DISTINCT	중복되지 않게 검색
WHERE 조건	조건식에 따라 검색
GROUP BY 컬럼명	컬럼값에 따라 그룹화
HAVING 조건	그룹별 조건
ORDER BY 컬럼명	컬럼명 순으로 정렬 (ASC 기본값)

SELECT 문 연습 위한 테이블 – DB명 COMPANY

employee

emp_no	name	department	position	gender	hire_date	Salary
Varchar(5)	Varchar(20)	Varchar(2)	Varchar(10)	Varchar(1)	Date	Int
1001	구창민	1	과장	М	1995-05-01	5000000
1002	김민서	1	사원	М	2017-09-01	2500000
1003	이은영	2	부장	F	1990-09-01	5500000
1004	한성일	2	과장	М	1993-04-01	5000000

department

dept_no	dept_name	location
Varchar(2)	Varchar(20)	Varchar(20)
1	영업부	대구
2	인사부	서울
3	총무부	대구
4	기획부	서울

```
모든 컬럼을 검색하고 싶다면 * 를 사용한다.

SELECT * FROM employee; employee테이블에서 모든 컬럼 검색 모든 테이블 내용이 출력
```



조건에 맞는 레코드 중 컬럼 출력

SELECT <u>name</u> name 컬럼만 출력

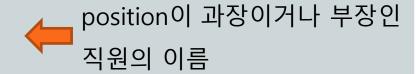
FROM employee ← employee테이블에서

WHERE *position = '과장'* position이 과장인 직원 검색

and(조건 2개 만족), or, not 을 써서 조건식을 만들 수 있다.

SELECT <u>name</u>

FROM employee



WHERE position = '과장' OR position = '부장'

급여가 5,000,000 이상인 직원의 사원번호와 이름을 검색 하시오.



남자 과장의 채용일을 검색하시오.

MariaDB [olympicsdb]> selec	t hire_d	ate from	employee w	vhere ger	nder="M"	and positi	on='과장';
 hire_date	emp_no	name	department	position	gender	hire_date	Salary
1995-05-01	1001	구창민	1	과장	М	1995-05-01	5000000
1993-04-01	1002	김민서	1	사원	М	2017-09-01	2500000
2 rows in set (0.001 sec)	1003	이은영	2	부장	F	1990-09-01	5500000
	1004	한성일	2	과장	М	1993-04-01	5000000

우리회사가 위치한 지역을 검색하시오. 중복되는 것은 하나만

MariaDB [olympicsdb]> select location f ++ location	rom departi	ment;	
++	dept_no	dept_name	location
대구 서울 대구 서울	1	영업부	대구
[대충	2	인사부	서울
서울 ++	3	총무부	대구
4 rows in set (0.000 sec)	4	기획부	서울



```
SELECT 컬럼1, 컬럼2, ...
FROM 테이블_이름
```

ORDER BY 컬럼1, *컬럼*2,

컬럼1, 컬럼2 순으로 정렬하여 출력

ASC(ASCENDING) : 오름차순

DESC(DESCENDING) : 내림차순

ASC나 DESC 생략하는 경우 ASC로 정렬

SELECT *

FROM employee

월급이 많은 순서로 직원의 모든 정보 출력

ORDER BY salary DESC;

WHERE 에 IN 사용하기

OR조건을 여러 개 쓰는 것과 같음. 지정된 범위 중 일치하는 값이 하나라도 있으면 검색 예) position in ('과장','부장') => 직급이 과장이거나 부장인

직급이 과장, 부장인 사람의 이름을 검색하세요.

SELECT <u>name</u>

FROM employee

WHERE position = '과장' OR position = '부장'

2개의 결과는 같음.

SELECT *name*

FROM employee

WHERE position in ('과장' ,'부장');

WHERE 에 IN 사용하기

In 연산자의 장점

조건 값이 여러 개 일 때 사용하기 쉽다.

Or연산자보다 실행속도가 빠르다.

In 연산자에 다른 select문을 넣을 수 있다.

emp_no	name	department	position	생략
1001	구창민	1	과장	
1002	김민서	1	사원	
1003	이은영	2	부장	
1004	한성일	2	과장	

대구에 있는 지점에 근무하는 사원의 이름과 부서코드, 직급을 검색하세요.

SELECT <u>name</u>, <u>department</u>, <u>position</u>

FROM employee

WHERE department in (

dept_no	dept_name	location
1	영업부	대구
2	인사부	서울
3	총무부	대구
4	기획부	서울

select dept_no from department where location='대구');

1차로 수행 된 결과. Where department in ('1', '2')

WHERE 에 BETWEEN 사용하기

And 연산과 동일, 특정 범위 값을 검색할 때 사용예) salary >= 3000000 and salary <=5000000 salary between 3000000 and 5000000

급여가 3,000,000 이상 5,000,000 이하인 직원의 이름을 검색하세요.

SELECT *name*

FROM employee

WHERE salary >= 3000000 and salary <= 5000000;

2개의 결과는 같음.

SELECT *name*

FROM employee

WHERE salary between 3000000 and 5000000;

SELECT 문(다양한 함수)



SELECT 문(다양한 함수)

숫자에 콤마 찍기 format()

SELECT FORMAT(컬럼명,0) FROM 테이블_ 이름

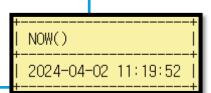
select format(min(salary),0) from employee;



현재 날짜 보기 now()

SELECT NOW() FROM 테이블_ 이름

select distinct now() from employee;



문자열 합쳐서 출력 concat(문자열,문자열,..)

SELECT CONCAT(문자열, 문자열,...) FROM 테이블_ 이름

select concat(emp_no, "-", name) from employee;



SELECT 문 - GROUP BY, HAVING

GROUP BY - 조회 결과를 그룹으로 묶어서 결과를 가져온다.

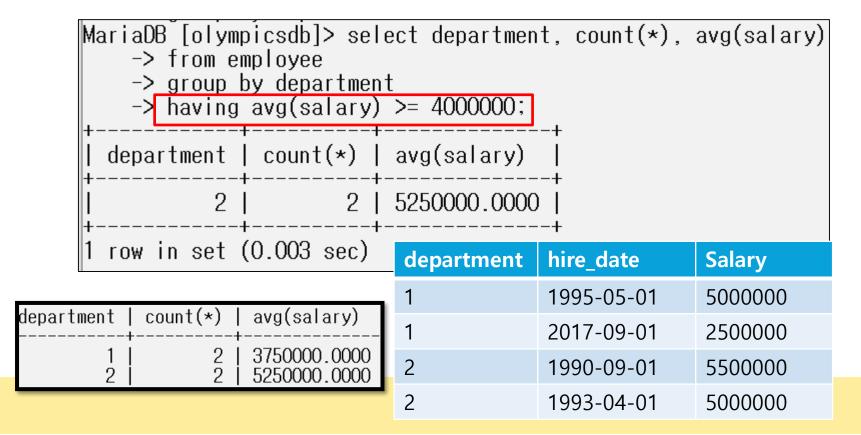
부서별로 직원수와 평균 급여를 구하시오.

department	hire_date	Salary
1	1995-05-01	5000000
1	2017-09-01	2500000
2	1990-09-01	5500000
2	1993-04-01	5000000

SELECT 문 - GROUP BY, HAVING

HAVING – GROUP BY 절에 조건문을 쓸 때 사용한다.

부서별로 직원수와 평균 급여를 구하고 <u>평균 급여가 4000000 이상</u>인 곳을 검 색하시오.



SELECT 문(LIKE)

특정 패턴의 값을 찾을 때 쓰는 연산자 LIKE

```
SELECT 컬럼1, 컬럼2, ... FROM 테이블_이름
WHERE 컬럼명 LIKE 패턴;
```

```
와일드 카드 문자
%:여러 글자 _ : 한 개의 글자
```

select name, department, position from employee where name like '윤%'; select name, department, position from employee where name like '%윤%'; select name, department, position from employee where name like '_윤%'; select name, department, position from employee where name like '_윤_'; select name, department, position from employee where name like '%윤';

```
순서대로 name에(이)
```

- * '윤'으로 시작하는 사람 * '윤'이라는 글자가 들어가는 사람,
- * 두번째 글자가 '윤'인 사람 * 3글자중 2번째 글자가 '윤'인 사람
- * '윤 ' 으로 끝나는 사람

UPDATE ~ SET 문

테이블에 저장된 값을 수정

UPDATE 테이블 이름

SET 컬럼1 = 값1, 컬럼2 = 값2, ...

WHERE 조건식;

UPDATE *employee*

SET position = '대리'

WHERE emp_name = '김민서';

김민서 사원의 직급이

사원에서 대리로 변경

UPDATE employee

SET salary = salary + 1000

WHERE department = 2,

부서코드가 2인 부서원

의 월급을 1000원 올려

준다.

DELETE FROM 문

테이블에 저장된 값을 삭제

주의 : 요기에 * 없음.

DELETE FROM 테이블_이름 WHERE 조건식;

DELETE FROM employee WHERE emp_no=1004;

emp_no가 1004인 직원 정보 삭제

DELETE FROM employee

모든 테이블 정보 삭제

주의!! 테이블의 모든 내용이 삭제되므로 주의해서 사용.