SQL 기초 Syntax

학습목표

1. SQL의 기본적인 사용법을 학습 합니다.

1. SELECT

SELECT의 안 좋은 습관

```
SELECT *
FROM    sample.sample_employee
;
```

“USE” 문을 명시적으로 사용합니다.

“*” 사용 하지 않도록 합니다.

SELECT의 바람직한 예

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM    sample_employee
;
```

“USE”을 명시적으로 사용합니다.

“*” 를 사용하지 않고 column 명을 명시적으로 사용합니다.

SELECT 특정 열 선택

```
USE sample;

SELECT employee_num
      , employee_name
FROM    sample_employee
;
```

산술 연산자와 NULL

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price, sales_price + 1000
      , insert_date
FROM    sample_sales
;
```

sales_price 에는 NULL 값이 있습니다.

NULL 값이 산술 연산자와 만나면 어떠한 상황이 되는지 확인 합니다.

산술 연산자 우선 순위

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price, 2*sales_price+1000, 2*(sales_price+1000)
      , insert_date
FROM    sample_sales
;
```

*결과를 조회 하여 산술 연산자의 우선 순위를 확인 합니다.

0으로 나누기

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price, 2 / sales_price
      , insert_date
FROM    sample_sales
;
```

.

별칭 사용

```
USE sample;

SELECT employee_num AS NUM
      , trade_kind    AS TR_CD
      , sales_price   AS PRICE, 2 / sales_price AS DIV_ZERO
      , insert_date   AS INT_DT
FROM    sample_sales
;
```

연결 연산자

```
USE sample;

SELECT employee_num
      , CONCAT(employee_name,"의 전화 번호는 ", contact_number," 입니다.")
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM    sample_employee
;
```

.

중복행 제거

```
USE sample;

SELECT employee_num
FROM    sample_sales
;

SELECT DISTINCT employee_num
FROM    sample_sales
;
```

2. 데이터 제한 및 설정

WHERE 절 사용 숫자

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM   sample_employee
WHERE  employee_num = 2
;
```

WHERE 절 사용 문자

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM   sample_employee
WHERE  employee_name = '김길동2'
;
```

WHERE 절 사용 비교(1)

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   sales_price >= 95000
;
```

WHERE 절 사용 비교(2)

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   sales_price <= 3000
;
```

WHERE 절 사용 비교(3)

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   sales_price = 98024
;
```

.

WHERE 절 사용 비교(4)

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   sales_price <> 98024
;
```

.

WHERE 절 사용 BETWEEN

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   sales_price BETWEEN 94000 AND 96000
;
```

WHERE 절 사용 IN

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   insert_date IN ('2015-01-01', '2015-02-01', '2015-03-01')
;
```

.

WHERE 절 사용 LIKE

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM    sample_employee
WHERE   contact_number LIKE '016%'
;
```

LIKE 사용 시 %검색조건% 형태의 쿼리는 사용하지 마시길 바랍니다.

WHERE 절 사용 ESCAPE LITERALS

```
USE sample;

SELECT escape_literals
FROM    sample_escape_literals
;

SELECT escape_literals
FROM    sample_escape_literals
WHERE   escape_literals LIKE '016₩%%'
;
```

특수기호나 예약어 제외 시 사용합니다.

NULL 검색

```
USE sample;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  sales_price = NULL
;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  sales_price <> NULL
;
```

```
USE sample;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  sales_price IS NULL
;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  sales_price IS NOT NULL
;
```

AND 연산자

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   employee_num = 10
AND     sales_price < 4000
;
```

OR 연산자

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   employee_num = 10
OR      sales_price < 4000
;
```

.

NOT 연산자

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
WHERE   employee_num   NOT IN ( 10, 20)
;
```

.

조회 시 연산자 우선 순위

```
USE sample;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  sales_price   > 98000
OR     employee_num  = 20
AND    sales_price   < 4000
;
```

```
USE sample;

SELECT employee_num
     , trade_kind
     , sales_price
     , insert_date
FROM   sample_sales
WHERE  (sales_price  > 98000
OR     employee_num  = 20 )
AND    sales_price   < 4000
;
```

***결과를 조회 하여 조회 시 연산자의 우선 순위를 확인 합니다.**

ORDER BY 절

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
ORDER BY insert_date ASC
;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
ORDER BY insert_date DESC
;
```

별칭 ORDER BY 절

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price * 2 AS S_PRICE
      , insert_date
FROM    sample_sales
ORDER BY S_PRICE DESC
;
```

복수 ORDER BY 절

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , insert_date
FROM    sample_sales
ORDER BY insert_date DESC, employee_num DESC
;
```

3. 함수

대(소)문자 조작 함수(1)

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email                 , LOWER(email), UPPER(email),
      , senior_manager_num
      , contract_date
      , update_date
FROM    sample_employee
;
```

.

대(소)문자 조작 함수(2)

```
USE sample;

SELECT employee_num
      , employee_name
      , contact_number
      , email
      , senior_manager_num
      , contract_date
      , update_date
FROM    sample_employee
WHERE   email = 'AK3@TMON.CO.KR'
;
```

*조회를 해서 조회 값이 맞는지 확인 합니다.

문자 열 조작 함수

```
USE sample;

SELECT employee_num
      , CONCAT(employee_name," : ",email) AS NAME_EMAIL
      , CHAR_LENGTH(employee_name)         AS NAME_LENG
      , SUBSTR(contact_number, 1,  3)      AS PHONE1
      , SUBSTR(contact_number, 5,  4)      AS PHONE2
      , SUBSTR(contact_number, 10, 4)      AS PHONE3
      , email
      , INSTR(email, '@')                  AS POS_DELIMT
      , LPAD(SUBSTR(email, INSTR(email, '@'),11), 14, '*') AS LPAD_EMAIL
      , RPAD(SUBSTR(email, 1, INSTR(email, '@')), 14, '*') AS RPAD_EMAIL
FROM    sample_employee
;
```

각각의 함수가 하는 역할을 확인 합니다.

숫자 조작 함수

```
USE sample;

SELECT ROUND(45.923,2), ROUND(45.923,0), ROUND(45.923,-1);

SELECT TRUNCATE(45.923,2), TRUNCATE(45.923, 0), TRUNCATE(45.923,-2);

SELECT employee_num
      , trade_kind
      , sales_price, MOD(sales_price, 100)
      , insert_date
FROM    sample_sales
;
```

각각의 함수가 하는 역할을 확인 합니다.

날짜 함수

```
USE sample;

SELECT CURDATE();

SELECT SYSDATE();

SELECT NOW();

SELECT DATEDIFF( CURDATE(), CONVERT('2014-11-21', DATE) );

SELECT DATE_ADD(CONVERT('2014-11-21', DATE), INTERVAL 63 DAY);

SELECT DATE_ADD(CONVERT('2014-11-21', DATE), INTERVAL 2 MONTH);

SELECT DATE_ADD( CURDATE(), INTERVAL 1 DAY);

SELECT LAST_DAY( CURDATE() );
```

각각의 함수가 하는 역할을 확인 합니다.

데이터의 암시적 변환

```
USE sample;

SELECT DATEDIFF( CURDATE(), '2014-11-21' );

SELECT '2' - 1;

SELECT CONCAT( CURDATE(), ' 00:00:00');
```

어느 부분이 암시적 변환인지 확인합니다.

데이터의 명시적 변환

```
USE sample;

SELECT DATEDIFF( CURDATE(), CAST('2014-11-21' AS DATE) );

SELECT CONVERT('2', UNSIGNED INTEGER) - 1;

SELECT CONCAT( CAST(CURDATE() AS CHAR(10)), ' 00:00:00');
```

날짜 형식

```
USE sample;

SELECT DATE_FORMAT(CONVERT('2014-11-21 15:01:23', DATETIME), '%Y-%m-%d %h:%m:%s');

SELECT DATE_FORMAT(CONVERT('2014-11-21 15:01:23', DATETIME), '%D %M %Y (%a) %h:%m:%s');
```

***다양한 DATE_FORMAT을 확인합니다.**

NULL 치환 함수

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , COALESCE( sales_price , 0, sales_price)
      , insert_date
FROM    sample_sales
;


SELECT employee_num
      , trade_kind
      , sales_price
      , IFNULL( sales_price, 0)
      , insert_date
FROM    sample_sales
;
```

치환 함수

```
USE sample;

SELECT employee_num
      , trade_kind
      , sales_price
      , IF( sales_price IS NULL, 0, sales_price)
      , insert_date
FROM    sample_sales
;


SELECT employee_num
      , trade_kind
      , CASE WHEN sales_price IS NULL THEN 0 ELSE sales_price END
      , insert_date
FROM    sample_sales
;
```

4. JOIN

CARTESIAN PRODUCT

```
USE sample;

SELECT COUNT(*)
FROM    sample_employee
;

SELECT COUNT(*)
FROM    sample_sales
;
```

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SS.employee_num
      , SS.trade_kind
      , SS.sales_price
      , SS.insert_date
FROM    sample_employee SE INNER JOIN sample_sales
SS
;

SELECT COUNT(*)
FROM    sample_employee SE INNER JOIN sample_sales
SS
;
```

등가 조인(1)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SS.employee_num
      , SS.trade_kind
      , SS.sales_price
      , SS.insert_date
FROM    sample_employee SE INNER JOIN sample_sales SS ON SE.employee_num = SS.employee_num
;
```

sample_empolyee 테이블과 sample_salses 는 1:N의 관계 입니다.

등가 조인(2)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SS.employee_num
      , SS.trade_kind
      , SS.sales_price
      , SS.insert_date
      , SEN.employee_num
      , SEN.nationality
FROM    sample_employee SE INNER JOIN sample_sales SS                    ON SE.employee_num =
SS.employee_num
                        INNER JOIN sample_employee_nationality_history SEN ON SE.employee_num =
SEN.employee_num
;
```

sample_empolyee 테이블과 sample_employee_nationality_history 는 1:N의 관계 입니다.

등가 조인(3)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SS.employee_num
      , SS.trade_kind
      , SS.sales_price
      , SS.insert_date
      , SEN.employee_num
      , SEN.nationality
FROM    sample_employee SE INNER JOIN sample_sales SS                    ON SE.employee_num =
SS.employee_num
                        INNER JOIN sample_employee_nationality_history SEN ON SE.employee_num =
SEN.employee_num
WHERE   SE.employee_num = 30
;
```

비등가 조인

USE sample;

```
SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SS.employee_num
      , SS.trade_kind
      , SS.sales_price
      , SS.insert_date
FROM    sample_employee SE INNER JOIN sample_sales SS ON SE.employee_num = SS.employee_num
                                                    AND SS.sales_price BETWEEN 1 AND 10000
;
```

포괄 조인(LEFT)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SEN.employee_num
      , SEN.nationality
FROM   sample_employee SE LEFT JOIN sample_employee_nationality_history SEN ON SE.employee_num =
SEN.employee_num
;
```

포괄 조인(RIGHT)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SEN.employee_num
      , SEN.nationality
FROM   sample_employee SE RIGHT JOIN sample_employee_nationality_history SEN ON SE.employee_num =
SEN.employee_num
;
```

***결과를 확인합니다.**

자체 조인(SELF)

```
USE sample;

SELECT SE.employee_num
      , SE.employee_name
      , SE.contact_number
      , SE.email
      , SE.senior_manager_num
      , SE.contract_date
      , SE.update_date
      , SE2.employee_num
      , SE2.employee_name
FROM   sample_employee SE INNER JOIN sample_employee SE2 ON SE.senior_manager_num = SE2.employee_num
;
```

5. 그룹 함수

AVG 및 SUM

```
USE sample;

SELECT AVG(sales_price)
      , MAX(sales_price)
      , MIN(sales_price)
      , SUM(sales_price)
FROM    sample_sales
WHERE   employee_num = 10
;


SELECT MAX(insert_date)
      , MIN(insert_date)
FROM    sample_sales
WHERE   employee_num = 10
;
```

AVG는 NULL 값을 포함하지 않습니다.

MAX와 MIN은 숫자 뿐 아니라 날짜도 처리 할 수 있습니다.

NULL을 포함한 AVG 처리를 확인 합니다.

COUNT

```
USE sample;

SELECT COUNT(sales_price)
FROM   sample_sales
WHERE  employee_num = 10
;


SELECT COUNT(sales_price)
FROM   sample_sales
WHERE  employee_num = 10
AND    sales_price IS NULL
;
```

COUNT는 NULL 값을 포함하지 않습니다.

NULL을 COUNT 확인 합니다.

중복 제거

```
USE sample;

SELECT DISTINCT employee_num
FROM    sample_sales
;
```

DISTINCT는 NULL를 처리 합니다.

중복 제거

```
USE sample;

SELECT DISTINCT employee_num
FROM    sample_sales
;
```

DISTINCT는 NULL를 처리 합니다.

GROUP BY

```
USE sample;

SELECT employee_num, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num
;
```

GROUP BY는 NULL를 처리 합니다.

GROUP BY 정렬

```
USE sample;

SELECT employee_num, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num
ORDER BY AVG(sales_price) DESC
;
```

GROUP BY는 NULL를 처리 합니다.

복수 행 GROUP BY

```
USE sample;

SELECT employee_num, trade_kind, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num, trade_kind
ORDER BY AVG(sales_price) DESC
;
```

GROUP BY는 NULL를 처리 합니다.

복수 행 GROUP BY

```
USE sample;

SELECT employee_num, trade_kind, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num, trade_kind
;
```

GROUP BY는 NULL를 처리 합니다.

HAVING 절

```
USE sample;

SELECT employee_num, trade_kind, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num, trade_kind
HAVING AVG(sales_price) > 45000
ORDER BY AVG(sales_price) DESC
;
```

GROUP BY는 NULL를 처리 합니다.

그룹 함수의 중첩

```
USE sample;

SELECT MAX(AVG(sales_price))
FROM   sample_sales
GROUP BY employee_num
;
```

GROUP_CONCAT

```
USE sample;

SELECT employee_num, GROUP_CONCAT(DISTINCT trade_kind SEPARATOR'+') AS exps, AVG(sales_price)
FROM    sample_sales
GROUP BY employee_num
;
```

NULL값을 포함한 정상금액(trade_kind=1)에서 비정상 금액(trade_kind =0)을 뺀 AVG 금액을 산출해 봅니다.

6. 서브 쿼리

단일행 서브 쿼리

```
USE sample;

SELECT employee_num
     , employee_name
     , contact_number
     , email
     , senior_manager_num
     , contract_date
     , update_date
FROM   sample_employee
WHERE  employee_num = (
                       SELECT employee_num
                       FROM   sample_sales
                       WHERE  sales_price > 99800
                      )
;
```

다중행 서브 쿼리

```
USE sample;

SELECT employee_num
     , employee_name
     , contact_number
     , email
     , senior_manager_num
     , contract_date
     , update_date
FROM   sample_employee
WHERE  employee_num IN (
                        SELECT employee_num
                        FROM   sample_sales
                        WHERE  sales_price > 99000
                       )
;
```

```
USE sample;

SELECT employee_num
     , employee_name
     , contact_number
     , email
     , senior_manager_num
     , contract_date
     , update_date
FROM   sample_employee
WHERE  employee_num = ANY (
                        SELECT employee_num
                        FROM   sample_sales
                        WHERE  sales_price > 99000
                       )
;
```

7. DML

INSERT

USE sample;

```
INSERT INTO sample_employee (employee_num, employee_name, contact_number, email, senior_manager_num,
contract_date, update_date)
    VALUES(100 , '김철수' , '010-1234-5678', 'kcs@tmon.co.kr', 20, CURDATE(), CURDATE())
;

INSERT INTO sample_employee VALUES(102, '김철수', '010-1234-5678', 'kcs@tmon.co.kr', 20, CURDATE(), CURDATE())
;

INSERT INTO sample_employee VALUES
 (104 , '김철수'  , '010-1234-5678'  , 'kcs@tmon.co.kr'  , 20  , CURDATE()  , CURDATE())
,(106 , '김철수'  , '010-1234-5678'  , 'kcs@tmon.co.kr'  , 20  , CURDATE()  , CURDATE())
;
```

INSERT NULL

```
USE sample;

INSERT INTO sample_employee VALUES
 (108 , '김철수'  , '010-1234-5678' , NULL , NULL  , CURDATE()  , CURDATE())
,(110 , '김철수'  , '010-1234-5678' , ''   , NULL  , CURDATE()  , CURDATE())
;
```

결과를 조회 해서 NULL값과 NULL이 아닌 값을 확인합니다.

테이블을 이용한 INSERT

```
USE sample;


INSERT INTO sample_employee
SELECT NULL
      , employee_name
      , contact_number
      , NULL
      , NULL
      , CURDATE()
      , CURDATE()
FROM    sample_employee
WHERE   employee_num = 60
;
```

***조회 해서 employee_num 값을 확인 합니다.**

UPDATE

USE sample;

```
UPDATE sample_employee
SET     email = 'kkk@yahoo.co.kr'
WHERE   employee_num = 60
;

UPDATE sample_employee
SET     email = 'kkk@yahoo.co.kr'
;
```

테이블을 이용한 UPDATE

```
USE sample;

UPDATE sample_employee
SET    senior_manager_num = (
                             SELECT senior_manager_num
                             FROM   sample_employee
                             WHERE  employee_num = 60
                            )
WHERE  senior_manager_num IS NULL
;
```

***employee_num=60의 senior_manager_num 값을**

senior_manager_num 가 NULL 데이터에 UPDATE 처리 합니다.

DELETE

USE sample;

```
DELETE
FROM    sample_employee
WHERE   employee_num =   100
;
```

테이블을 이용한 DELETE

```
USE sample;


DELETE
FROM    sample_employee
WHERE   employee_num IN (
                        SELECT employee_num
                        FROM    sample_employee
                        WHERE   senior_manager_num = 20
                        AND     employee_num > 100
                       )
;
```

***senior_manager_num =20 이면서 employee_num > 100**

데이터를 삭제 합니다.

REPLACE

USE sample;

```
REPLACE INTO sample_employee ( employee_num, employee_name, contact_number, email, senior_manager_num,
contract_date, update_date )VALUES
 (58 , '김철수' , '010-1234-5678' , 'kcs@tmon.co.kr' , 20 , CURDATE() , CURDATE())
,(100 , '김철수2' , '010-1234-5678' , 'kcs@tmon.co.kr' , 20 , CURDATE() , CURDATE())
;
```

***쿼리문에 의해 영향 받은 ROW 개수를 확인합니다.**

명시적 TRANSACTION DML

```
USE sample;

SET SESSION autocommit = 'off';

INSERT INTO sample_employee
SELECT NULL
      , employee_name
      , contact_number
      , NULL
      , NULL
      , CURDATE()
      , CURDATE()
FROM    sample_employee
WHERE   employee_num = 60
;

SAVEPOINT A;
```

```
USE sample;

INSERT INTO sample_employee
SELECT NULL
      , employee_name
      , contact_number
      , NULL
      , NULL
      , CURDATE()
      , CURDATE()
FROM    sample_employee
WHERE   employee_num = 60
;

ROLLBACK TO A;

ROLLBACK;

SET SESSION autocommit = 'on';
```

8. 테이블 생성 관리

암시적 TRANSACTION DDL

USE sample;

```
DROP TABLE IF EXISTS department;
CREATE TABLE department(
  department_cd    INT(11)      NOT NULL AUTO_INCREMENT COMMENT '부서번호'
, department_name VARCHAR(10) NOT NULL                  COMMENT '부서이름'
, PRIMARY KEY (department_cd)
)ENGINE=INNODB CHARSET=utf8 COMMENT='부서테이블'
;

SHOW TABLES;

SHOW CREATE TABLE department;
```

테이블 생성 방법

```
USE sample;

DROP TABLE IF EXISTS department_2;
CREATE TABLE department_2 LIKE department;


DROP TABLE IF EXISTS department_2;
CREATE TABLE department_2
SELECT *
FROM    department
;

DROP TABLE IF EXISTS department_2;
CREATE TABLE department_2
SELECT department_cd
      , department_name
      , CAST( NULL AS DATETIME) AS insert_date
FROM    department
;

SHOW CREATE TABLE department_2;
```

테이블 컬럼 변경 방법

USE sample;

ALTER TABLE sample_employee ADD COLUMN test1 CHAR(1) NOT NULL DEFAULT 'Y' COMMENT '테스트' AFTER update_date;

ALTER TABLE sample_employee CHANGE COLUMN test1 test2 CHAR(1) NOT NULL DEFAULT 'Y' COMMENT '테스트' AFTER update_date;

ALTER TABLE sample_employee MODIFY COLUMN test2 VARCHAR(10) NULL COMMENT '테스트' AFTER update_date;

ALTER TABLE sample_employee DROP COLUMN test2;

***다양한 데이터 타입에 대해 확인 합니다.**

테이블 이름 및 속성 변경 방법

```
USE sample;

ALTER TABLE department RENAME department_3;

RENAME TABLE department_3 TO department_4;


ALTER TABLE department_4 COMMENT ='부서테이블 이름 변경';

ALTER TABLE department_4 CONVERT TO CHARACTER SET euckr;

ALTER TABLE department_4 CHARACTER SET utf8;

ALTER TABLE department_4 ENGINE='myisam';


SHOW CREATE TABLE department_4;
```

결과 테이블 속성을 조회 합니다.

테이블 삭제 방법

```
USE sample;

TRUNCATE TABLE department_4;

DROP TABLE IF EXISTS department_4;
```

9. 제약 조건

NOT NULL

```
USE sample;

INSERT INTO sample_employee (employee_num, employee_name, contact_number, email, senior_manager_num,
contract_date , update_date)
 VALUES (62  ,'김길동31', NULL , 'ck12@tmon.co.kr' , 20 , '2014-04-07' , '2014-04-07')
;

INSERT INTO sample_employee (employee_num, employee_name, contact_number, email, senior_manager_num,
contract_date , update_date)
 VALUES (62  ,'김길동31', DEFAULT , 'ck12@tmon.co.kr' , 20 , '2014-04-07' , '2014-04-07')
;

INSERT INTO sample_employee (employee_num, employee_name, contact_number, email, senior_manager_num,
contract_date , update_date)
 VALUES (64  ,'김길동31', NULL , 'ck12@tmon.co.kr' , 20 , '2014-04-07' , '2014-04-07')
       ,(66  ,'김길동31', NULL , 'ck12@tmon.co.kr' , 20 , '2014-04-07' , '2014-04-07')
;
```

***Multi INSERT 시 NOT NULL 조건이 처리 되는 방법을 확인 합니다.**

UNIQUE KEY

```
USE sample;

SELECT    COUNT(mapping_id)
FROM      sample_department_mapping
GROUP BY employee_security_code
HAVING    COUNT(*) > 1
;

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code)
     VALUES(35, 16, 70, 'XDFE034')
;

ALTER TABLE sample_department_mapping MODIFY employee_security_code CHAR(7) NULL UNIQUE COMMENT '사
원보안코드';

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code)
VALUES
   (35, 16, 70, NULL)
  ,(36, 16, 70, NULL)
;
```

UNIQUE 가 NOT NULL 이여야만 하는 이유를 확인합니다.

PRIMARY KEY

```
USE sample;

ALTER TABLE sample_department_mapping DROP PRIMARY KEY;

ALTER TABLE sample_department_mapping
    MODIFY COLUMN  mapping_id    INT(11)  NOT NULL COMMENT '매핑ID'
  , DROP PRIMARY KEY
;

ALTER TABLE sample_department_mapping
    MODIFY COLUMN  mapping_id    INT(11) NOT NULL AUTO_INCREMENT COMMENT '매핑ID'
  , ADD PRIMARY KEY (mapping_id)
;

SHOW CREATE TABLE sample_department_mapping;
```

***MySQL에서 PRIMARY KEY 및 AUTO_INCREMENT를 권장하는 이유를 확인 합니다.**

FOREIGN KEY (ON DELETE CASCADE)

```
USE sample;

INSERT INTO sample_department ( department_cd, department_name ) VALUES
    ( 22, '비서실');

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code )
VALUES
    ( 37, 24, 48, 'XDFE035')
;

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code )
VALUES
    ( 37, 22, 48, 'XDFE035')
;

DELETE
FROM     sample_department
WHERE    department_cd = 22
;
```

Sample_department_mapping 테이블을 조회하여 동작 결과를 확인합니다.

FOREIGN KEY (ON UPDATE CASCADE)

```
USE sample;


ALTER TABLE sample_department_mapping DROP FOREIGN KEY fk_department_cd
;

ALTER TABLE sample_department_mapping ADD CONSTRAINT fk_department_cd FOREIGN KEY (department_cd) REFERENCES
sample_department (`department_cd`) ON UPDATE CASCADE
;

INSERT INTO sample_department ( department_cd, department_name ) VALUES
    ( 22, '비서실');

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code ) VALUES
    ( 37, 22, 48, 'XDFE035')
;

UPDATE sample_department
SET    department_cd = 23
WHERE  department_cd = 22
;

DELETE
FROM    sample_department
WHERE   department_cd = 23
;
```

Sample_department_mapping 테이블을 조회하여 동작 결과를 확인합니다.

FOREIGN KEY (DISABLE)

```
USE sample;

DELETE
FROM     sample_department
WHERE    department_cd = 23
;
```

***sample_department 테이블의 데이터를 삭제할 수 있도록 합니다.**

ENUM

USE sample;

```
INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code,
use_yn ) VALUES
    ( 40, 20, 48, 'XDFE040', 1)
;

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code,
use_yn ) VALUES
    ( 41, 20, 48, 'XDFE041', 2)
;

INSERT INTO sample_department_mapping (mapping_id, department_cd, employee_num, employee_security_code,
use_yn ) VALUES
    ( 42, 20, 48, 'XDFE042', 3)
;
```

***sample_department_mapping 테이블을 조회하여 동작 결과를 확인합니다.**

10. VIEW

VIEW 생성 및 변경

USE sample;

```
CREATE OR REPLACE DEFINER=root@localhost VIEW v_sample_employee AS
SELECT    A.employee_num
        , A.employee_name
        , A.contact_number
        , A.email
        , A.senior_manager_num
        , A.contract_date
        , A.update_date
        , B.nationality
FROM      sample_employee A INNER JOIN sample_employee_nationality_history B ON A.employee_num = B.employee_num
;

SELECT    V.employee_num
        , V.employee_name
        , V.contact_number
        , V.email
        , V.senior_manager_num
        , V.contract_date
        , V.update_date
        , V.nationality
FROM      v_sample_employee V
;
```

VIEW를 통한 데이터 변경

USE sample;

```
INSERT INTO v_sample_employee (employee_num , employee_name , contact_number , email , senior_manager_num , contract_date ,
update_date) VALUES
    (62, '김길동31', '010-6545-9387', 'kkd31@tmon.co.kr', 16, CURDATE(), CURDATE())
;

UPDATE v_sample_employee
SET     contract_date = CURDATE()
     , update_date    = CURDATE()
WHERE   employee_num   = 32
;

DELETE
FROM    v_sample_employee2
WHERE   employee_num = 62
;
```

데이터 변경이 불가한 VIEW (1)

USE sample;

```
CREATE OR REPLACE DEFINER=root@localhost VIEW v_sample_employee AS
SELECT    A.employee_num
        , A.employee_name
        , A.contact_number
        , A.email
        , A.senior_manager_num
        , A.contract_date
        , A.update_date
        , B.nationality
FROM      sample_employee A LEFT INNER JOIN sample_employee_nationality_history B ON A.employee_num = B.employee_num
;

SELECT    V.employee_num
        , V.employee_name
        , V.contact_number
        , V.email
        , V.senior_manager_num
        , V.contract_date
        , V.update_date
        , V.nationality
FROM      v_sample_employee V
;
```

데이터 변경이 불가한 VIEW (2)

USE sample;

```
INSERT INTO v_sample_employee (employee_num , employee_name , contact_number , email , senior_manager_num , contract_date ,
update_date) VALUES
    (62, '김길동31', '010-6545-9387', 'kkd31@tmon.co.kr', 16, CURDATE(), CURDATE())
;


UPDATE v_sample_employee
SET     contract_date  = CURDATE()
     , update_date     = CURDATE()
WHERE   employee_num    = 34
;


DELETE
FROM    v_sample_employee
WHERE   employee_num = 60
;
```

***어떠한 VIEW가 데이터가 불가능한지 확인합니다.**

INLINE VIEW

USE sample;

```
SELECT    A.employee_num
        , A.employee_name
        , A.contact_number
        , A.email
        , A.senior_manager_num
        , A.contract_date
        , A.update_date
        , B.nationality
FROM      (
            SELECT   employee_num
                   , employee_name
                   , contact_number
                   , email
                   , senior_manager_num
                   , contract_date
                   , update_date
            FROM     sample_employee
            ORDER BY RAND()
            LIMIT 10
          ) A LEFT JOIN sample_employee_nationality_history B ON A.employee_num = B.employee_num
;
```

결과를 확인 하여 동작 방식을 확인합니다.

Q&A