1.기본 SQL 구문 파악
기본 SQL 정리 문서

1.1 DML
INSERT, DELETE, UPDATE, SELECT로 나뉘어 있음.

1.1.1 SELECT절
1.1.1.1 DESC 테이블명 : 테이블 구조 조회
1.1.1.2 기본검색 방법

```
SELECT 컬럼명
FROM 테이블명
WHERE 조건;

// 모든 컬럼을 출력할 경우 SELECT *
// 중복값을 제거할 경우 SELECT DISTINCT 컬럼명
//
```

1.1.2 WHERE절
1.1.2.1 기본 문법

```
SELECT 컬럼명
FROM 테이블명
WHERE 조건
ORDER BY [컬럼명, 표현식] [ASC / DESC];

//조건이 문자인 경우 WHERE NAME = 'TEAHYUN';
//조건이 날짜 데이터인 경우 WHERE DATE = #20200601#;
// BETWEEN 연산자 사용할 경우 WHERE 월급 BETWEEN 2500 AND 3500 ;
// IN연산자 사용할 경우 WHERE 컬럼명 IN(100,200,300);
//패턴 연산자 (_, %) 사용할 경우 WHERE 이름 LIKE'_ㅇ%'; <- 이름의 두번째 자리가 ㅇ로 시작하는 이름 출력 '%O%' <- O이 들어가는 모든 문자 출력
// IS NULL / IS NOT NULL 사용
WHERE 컬럼 IS NULL;
WHERE 컬럼 IS NOT NULL;
```

1.1.3 ORDER BY절
1.2.3.1 기본 문법
1.2.3.1.1 ASC : 오름차순
1.2.3.1.2 DESC : 내림차순

데이터 타입	ASC	DESC
숫자	0->9	9->0
날짜	이전 -> 이후	이후 -> 이전
문자	a -> z	z -> a

1.2.3.1.3 default : 오름차순
1.2.3.1.4 오름차순과 내림차순이 교집합일 때
우선 정렬은 왼쪽에 있는 컬럼이 우선 적용되며, 정렬 기준에서 같은 값을 가진 행이 있을 경우 후 순위의 정렬 기준이 적용된다.

1.1.4 GROUP BY절, HAVING 절
그룹함수, GROUP BY, HAVING

1.1.4.1 그룹함수
그룹단위로 값을 받아서 연산하는 함수 *(GROUP BY를 사용하지 않을 때, 하나의 테이블 = 하나의 그룹이 됨, NULL값을 가진 행은 연산에서 제외함, 중복값을 제거하는 DISTINCT도 사용 가능)

1.1.4.1.1 그룹함수 종류

숫자 값 n = 숫자값	AVG(ALL[DISTINCT] n)	평균값
	STDDEV(ALL[DISTINCT] n)	표준편차
	SUM(ALL[DISTINCT] n)	합계
	VARIANCE(ALL[DISTINCT] n)	분산
데이터 타입에 상관없이 사용 그룹 함수	COUNT(ALL[DISTINCT] *expr)	행 갯수
	MAX(ALL[DISTINCT] *expr)	max값을 구함
	MIN(ALL[DISTINCT] *expr)	min값을 구함

```
//사원테이블 전체 행 개수 출력
SELECT COUNT(*)
FROM 사원;
```

1.1.4.2 GROUP BY 절
GROUP BY절이란 ?
테이블의 행을 그룹으로 묶을 때 그룹을 묶는 기준을 설정하는 절.
기준 컬럼의 값이 동일한 행끼리 하나의 그룹으로 묶는다.
기준 컬럼의 값이 그룹의 이름이 된다.
쿼리구문에 WHERE절이 있는 경우 WHERE로 행을 먼저 선택한 뒤 그룹이 묶이게 된다.
그룹 함수를 사용한 경우 SELECT 절에는 그룹 함수, GROUP BY절에 명시된 컬럼만 올 수 있다.

1.1.4.2.1 기본 문법

```
SELECT 출력할 컬럼명, 그룹함수
FROM 검색할 테이블 명
WHERE 조건
GROUP BY 그룹으로 묶을 컬럼명
HAVING GROUP 조건
ORDER BY 정렬기준 컬럼명;
```

1.2.1 INSERT문
테이블에 새로운 행 정보를 입력할 때 사용하는 구문

1.2.1.1 기본문법

```
INSERT INTO 테이블명(컬럼1,컬럼2,)
VALUES(값1,값2);

예
INSERT INTO 학생(학과, 학번, 이름, 학년)
VALUES('컴퓨터공학','20201234','홍길동',1);
```

1.2.2 UPDATE문
기존 행의 정보를 갱신할 때 사용하는 구문

1.2.2.1 기본문법

```
UPDATE 테이블명
SET 수정할 컬럼명1 = 값1, 수정할 컬럼명2 = 값2
WHERE 조건;

//예제 학생테이블에서 이름이 홍길동인 학생의 학점을 'A'로 갱신
UPDATE 학생
SET 학점 = 'A'
WHERE 이름 = '홍길동';
```

1.2.3 DELETE문
기존 행의 정보를 삭제할 때 사용하는 구문

1.2.3.1 기본문법

```
DELETE FROM 테이블명
WHERE 조건 ;
```

1.3 CREATE문
새로운 테이블을 생성하는 구문
오브젝트(VIEW, INDEX, DATABASE) 등을 생성할 때도 사용

1.3.1 제약조건
테이블 생성 시, 각 컬럼에 제약조건을 설정하여 조건에 맞지 않는 데이터는 DB차원에서 입력되지 않도록 제한하는 조건

1.3.1.1 NOT NULL
컬럼에 NULL값을 허용하지 않음

1.3.1.2 UNIQUE
컬럼에 중복값을 허용하지 않음

1.3.1.3 PRIMARY KEY
중복값과 NULL값을 허용하지 않음
테이블 당 1번만 사용 가능
테이블을 대표하는 설정

1.3.2 예제
테이블명은 MEMBER테이블이다.
IDX는 정수형 기본키로 설정한다.
ID는 가변문자열형 10자리로 null값과 중복값을 가질 수 없다.
NAME은 가변문자열형 20자리이다.
EMAIL은 가변문자열형 30자리이고, 중복값을 가질 수 없다.
PHONE은 가변문자열형 20자리이고, NULL값과 중복값을 가질 수 없다.
START_DATE는 날짜형으로 지정한다.

```
CREATE TABLE MEMBER(
IDX INT CONSTRAINT MEM_ID_PK(생략) PRIMARY KEY,
ID VARCHAR(10) NOT NULL, UNIQUE,
NAME VARCHAR(20),
EMAIL VARCHAR(30) UNIQUE,
PHONE VARCHAR(20) NOT NULL, UNIQUE,
START_DATE DATE);
```

1.4 ALTER
테이블을 변경할 때 사용하는 구문

1.4.1 컬럼 추가

```
ALTER TABLE 테이블 이름
ADD (컬럼명 데이터타입(데이터크기));

//예제
학생 테이블에 학점 컬럼 추가
ALTER 학생
ADD (학점 VARCHAR(5));
```

1.4.2 컬럼 수정

```
ALTER TABLE 학생
MODIFY (학점 VARCHAR(10));
```

1.4.3 컬럼 삭제

```
ALTER TABLE 학생
DROP COLUMN 학점;
```

1.5 DROP문
테이블 삭제 명령어

1.5.1 기본문법

```
DROP TABLE 학생;

//테이블을 참조하는 외래키로 엮인 테이블 모두 삭제
DROP TABLE 학생 CASCADE CONSTRAINTS;
```

1.6 DDL(VIEW, INDEX), DCL(GRANT,REVOKE)

-

1.6.1 VIEW

물리적으로 존재하는 테이블이 아니라 논리적으로 존재하는 테이블

1.6.1.1 VIEW 생성

```
CREATE[OR REPLACE] [FORCE,NOFORCE] VIEW 뷰이름
AS 서브쿼리;
```

1.6.1.2 VIEW 삭제

```
DROP VIEW 뷰이름;
```

1.6.2 INDEX

-

1.6.2.1 INDEX생성

```
CREATE INDEX 인덱스명 ON 테이블명(인덱스 지정할 컬럼);
```

1.6.2.2 INDEX 삭제

VIEW와 같음.

```
DROP INDEX 인덱스명;
```

1.6.3 GRANT

DB를 조작할 수 있는 권한을 부여

1.6.3.1 예제

```
USER1에게 학생 테이블에 대한 select, insert, update, delete를 수행할 수 있는 권한을
부여 및 다른사람에게도 권한을 부여할 수 있도록 해줌.

GRANT SELECT, INSERT, UPDATE, DELETE ON 학생 TO USER1 WITH GRANT
OPTION;
```

1.6.4 REVOKE
권한을 회수

1.6.4.1 예제

USER1에게 부여한 학생테이블에 대한 DELETE 권한과 USER1이 타인에게 부여한 권한 모두 회수.

```
REVOKE DELETE ON 학생 FROM USER1 CASCADE CONSTRAINTS;
```