6장

SQL – RDB 언어

1>대소문자구분X(암호, '대소문자구분데이터' 예외)

2>문법 맞는 문장 실행

3>db 설치 컴퓨터 – db 설치 컴퓨터 네트워크 연결

SQL 종류

DQL	SELECT	조회
DDL	CREATE TABLE ALTER TABLE DROP TABLE	테이블 정의 수정 삭제
DML	INSERT UPDATE DELETE	테이블 내의 데이터 =레코드(여러 컬럼)저장 수정 삭제
DCL	GRANT REVOKE	DB 사용 권한 부여 회수 (단 DBA 만 사용 가능-system) hr 계정 명령어 사용불가
TCL	COMMIT ROLLBACK	트랜잭션 처리

SELECT 문법

SELECT	필수. 컬럼명,,,, * 연산식 별칭 distinct || 함수
FROM	필수. 테이블명
WHERE	생략가능. 조회 레코드 조건. 레코드 수 줄어든다
GROUP BY	생략가능. 집계함수 적용 그룹 컬럼명
HAVING	생략가능. 집계함수 조건식
ORDER BY	생략가능. 정렬 순서 컬럼명(index, 별칭) asc|desc , , ,

연산자

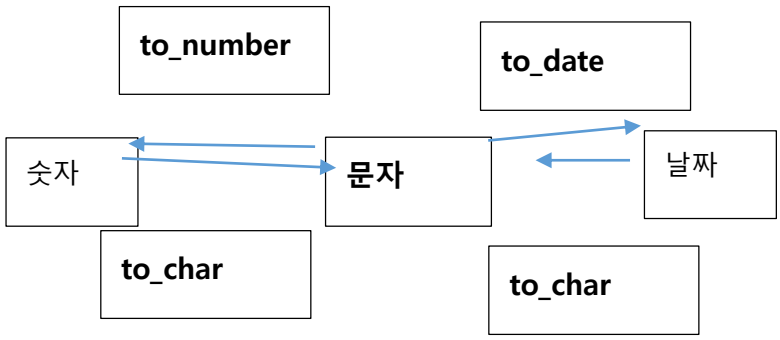
산술연산자	+ - * /
비교연산자	> >= < <= != =
조건결합 논리연산자	and or not
null 처리연산자	is null , is not null
목록 연산자	in (1, 2, 3, 4)
범위 연산자	between ? and ??
유사패턴비교	like '%' '_'

함수

집계함수	count(*) count(컬럼명) sum avg max min

문자형함수	length , lengthb , instr('java', 'a') -> 2 , substr ('java', 2, 1) -> 'a' upper lower initcap trim / ltrim / rtrim lpad / rpad
숫자형함수	abs 삼각함수 round-반올림 trunc –버림 mod- (자바 % 연산자 – 나머지 연산자 오라클 함수)
날짜형함수	1> + , - 연산 가능 (날짜 + 숫자(일))=> 날짜 리턴 TRUC, ROUND(날짜 + 날짜))=> 숫자(일단위) 리턴-> *24, /365 SYSDATE – 현재시각 표현 함수 . 'RR/MM/DD' MONTHS_BETWEEN (최근날짜1, 오래전날짜2) ADD_MONTHS(날짜, 개월숫자)
변환함수	NULL변환함수 – NVL(컬럼명, NULL값을 대체값)
	TO_CHAR – 날짜와 숫자를 문자로
	TO_NUMBER – 문자에서 숫자로
	TO_DATE – 문자에서 날짜로

- to_char / to_number / to_date



,	**, 기호**
$	**$ 기호**
L	**LOCALE CURRENCY - ₩**
9	**1자리숫자**
0	**1자리숫자. 0 표시**
YY – 2000년대	**년도**

YYYY **RR(0-49, 50-99)**	
MM	**월**
DD	**일**
HH **HH24**	**시간**
MI	**분**
SS	**초**
DAY	**요일**

- 02일 입사자

```
SELECT HIRE_DATE
FROM EMPLOYEES
WHERE TO_CHAR(HIRE_DATE, 'dd') = '02';
```

'02/08/13'

분석 순위함수- X

```
SELECT TO_CHAR(1000, '$9,999')  FROM DUAL;

SELECT  TO_NUMBER('$1,000' , '$9,999')  + 2000 FROM DUAL;

SELECT SYSDATE , TO_CHAR( SYSDATE , 'YYYY-MM-DD DAY HH24:MI:SS')     FROM DUAL;
```

$1,000
- 모든 사원의 COMMISSION_PCT 조회하되 NULL 사원은 0
(WHERE COMMISSION_PCT IS NULL ===> 공백)

```
SELECT NVL(COMMISSION_PCT , ?) FROM EMPLOYEES;
```

COMMISSION_PCT 타입 NUMBER(2,2) -> .14

- 사원 이름, 부서코드 DEPARTMENT_ID 조회하되 부서 없는 사원의 부서코드 '배치이전' 조회
NULL-숫자,날짜,문자컬럼

```
SELECT FIRST_NAME, NVL(TO_CHAR(DEPARTMENT_ID) , '배치이전')
```

FROM EMPLOYEES;

- 날짜 연산
SELECT SYSDATE-1 어제, SYSDATE 오늘, SYSDATE + 1 내일,
ADD_MONTHS(SYSDATE, 3) "3개월후" FROM DUAL;

-오늘 날짜와 입사일과의 경과일수 조회
SELECT SYSDATE - HIRE_DATE FROM EMPLOYEES
오늘 > 입사일
7007.43262--> 7007일 0.43262일 경과

-오늘 날짜와 입사일과의 경과 시간 조회
SELECT SYSDATE - HIRE_DATE FROM EMPLOYEES
7007.43262 * 24 --> 시간

-오늘 날짜와 입사일과의 경과년수 조회. 소수점 이하 버리자
SELECT TRUNC ((SYSDATE - HIRE_DATE) / 365) FROM EMPLOYEES
7007.43262 / 365

19.1984545 -> 19년. 0.1984545년

-오늘 날짜와 입사일과의 경과개월수 조회.

SELECT MONTHS_BETWEEN(SYSDATE, HIRE_DATE) FROM EMPLOYEES;

SELECT TRUNC ((SYSDATE - HIRE_DATE) / 365) FROM EMPLOYEES
7007.43262 / 365 --> 19년

-
-숫자 함수
select length('java') from dual;->4 (1행 함수 결과 가상 테이블 제공)
select length(first_name) from employees; -> 107행

select 34.5678 , trunc(34.5678), round(34.5678) from dual;
select 34.5678 , trunc(34.4678), round(34.4678) from dual;
select 34.5678 , trunc(34.4678 , 1), round(34.4678 , 1) from dual;
select 34.5678 , trunc(34.4678 , -1), round(34.4678 , -1) from dual;

```
select 34.5678 , trunc(34.4678, 0), round(34.4678, 0) from dual;
select 12345.6789, trunc(12345.6789 , -1), TRUNC(12345.6789 , -2), TRUNC(12345.6789 , -3)     from
dual;
```

trunc(숫자, X) / ROUND(숫자, X)
X-0 버림/반올림한 후에 정수 일자리까지 표현
X-양수 버림/반올림한 후에 소수점 자리까지 표현
X-음수 버림/반올림한 후에 정수 자리까지 표현

- 급여 평균 . 반올림하여 정수 표현

```
SELECT ROUND(AVG(SALARY)) FROM EMPLOYEES;

SELECT TRUNC(10 / 3) 정수몫, MOD(10, 3) 나머지 FROM DUAL;
```

- 사번 컬럼 EMPLPOYEE_ID 짝수인 사원의 사번, 이름 조회

```
SELECT EMPLOYEE_ID 사번 , FIRST_NAME 이름
FROM EMPLOYEES
WHERE MOD(EMPLOYEE_ID, 2) = 0;
```

사번 이름
100 XXXXX
102
104
....

11시 30분

데이터타입

날짜 – date –'rr/mm/dd'
문자 – varchar2(100) ' '
숫자 – number(5) 5자리 정수 number(5, 2) 3자리 정수와 2자리 소수점 이하

7.2 조인

join query

=

select

from 테이블명1

- 사원 이름, 부서코드 조회

```
select first_name, department_id
from employees;
```

- 부서코드와 부서이름 조회

```
select department_id, department_name
from departments;
```

- 사원 이름, 부서 이름 조회 – inner join

```
select first_name, employees.department_id, department_name
from departments , employees ;
```

```
select first_name, e.department_id, department_name
from departments d , employees e
where d.department_id = e.department_id; => 오라클 독자 문법 join
```

===>

```
select first_name, e.department_id, department_name
from departments d inner join employees e
on d.department_id = e.department_id; ==>ansi join (모든 db 표준 문법)
```

27 * 107 ==> 2889

employees	departments
William 10 Peter 20 King 30 Kimberly null	10 인사부 20 총무부 30 교육부

3*3=9

-join – 2 개 이상의 테이블을 동시 조회

1> select 컬럼명들
from 테이블명1, 2, ...
where 테이블명1, 2 컬럼값 "동일" 조건

2> 종류
inner join – 조건 만족 범위 내부 레코드 조인
outer join - 조건 만족 범위 외부 레코드 조인
self join – 자신 테이블 조인

- 사원 이름, 부서 이름 조회 – inner join
select first_name, e.department_id, department_name
from departments d , employees e
where d.department_id = e.department_id; -> 106
null(x) = null 값(킴벌리인 경우 존재)

- 사원 이름, 부서 이름 조회하되
부서코드 null 인 사원 포함 부서코드 '없음' 조회
select first_name, nvl(e.department_id, 0) , nvl(department_name, '없음')
from departments d , employees e
where d.department_id(+) = e.department_id; ->107
null(x) = null 값 포함(킴벌리인 경우 존재)

join시에 1개 테이블에 존재하는 값이 다른 테이블에 존재하지 않는 경우
join 포함x
--> (+) --> outer join

- 사원 이름, 부서 이름 조회하되
해당 부서 소속 사원이 없을 때 ㅖ 사원이름 '부서원없음' 조회

select nvl(first_name, '부서원없음') , d.department_id, department_name
from departments d , employees e
where d.department_id = e.department_id(+) ;
부서코드있다 = 부서 소속 부서원 없다(first_name null)
270 (o) = 270 (x)

departments 테이블 : 10 20 ... 270
270번 부서 속한 부서원 없다. = employees 테이블 부서코드 270 레코드 없다

self join - 상사 정보가 자신 테이블 포함

상사 없는 사원 포함 모든 사원의 상사 이름 급여 조회
select me.first_name 사원이름, me.manager_id, manager.first_name 상사이름, manager.salary
from employees me, employees manager
where me.manager_id=manager.employee_id(+);
null(o) null(x)

Jennifer 사원 포함 모든 사원의 상사 이름 급여 조회
select me.first_name 사원이름, me.manager_id, manager.first_name 상사이름, manager.salary
from employees me, employees manager
where me.manager_id=manager.employee_id
and me.first_name = 'Jennifer';

employees me			employees manager		
employee_id	first_name	manager_id	employee_id	first_name	manager_id
200	김사원	150	200	김사원	150
.....			.....		
150	최대리	100	150	최대리	100
....			....		
100	박과장	90	100	박과장	90

표준join – ansi	오라클 join
-inner join select employee_id, first_name, e.department_id, department_name from employees e inner join departments d on e.department_id = d.department_id;	-inner join select employee_id, first_name, e.department_id, department_name from employees e, departments d where e.department_id = d.department_id;
-outer join select employee_id, first_name, e.department_id, department_name from employees e left outer join departments	-outer join select employee_id, first_name, e.department_id, department_name from employees e, departments d

d
on e.department_id = d.department_id;
 	
select employee_id, first_name, e.department_id, department_name
from employees e right outer join departments d
on e.department_id = d.department_id;
	where e.department_id = d.department_id(+);
	
	select employee_id, first_name, e.department_id, department_name
	from employees e, departments d
	where e.department_id(+) = d.department_id;

SELECT me.first_name 내이름, me.salary 내급여, me.manager_id 상사사번, man.salary 상사급여 FROM EMPLOYEES me inner join EMPLOYEES man on me.manager_id=man.employee_id	-self join SELECT me.first_name 내이름, me.salary 내급여, me.manager_id 상사사번, man.salary 상사급여 FROM EMPLOYEES me, EMPLOYEES man WHERE me.manager_id=man.employee_id

2시 5분

join query / subquery

291p 7.2.5절 union union all not in in
6 장 any all
--->

select
from 테이블 여러개
where 여러개 테이블 결합 레코드 만드는 조건식

select (select ...)
from (select ...)
whre (select ...)

- Neena 의 급여 조회
select first_name, salary
from employees
where upper(first_name)=Upper('neena');

<where 절 포함>
- Neena와 같은 부서원의 이름, 급여 조회
select department_id, first_name, salary
from employees
where department_id = (select department_id from employees where first_name='Neena')

Neena 부서코드
select department_id
from employees
where first_name='Neena';

<select – select 포함>
select sum(salary) from employees; --> ok

select salary, sum(salary) from employees; --> error
107 1

select salary, (select sum(salary) from employees) from employees;
107 - 복사

<where 절 포함>
- Neena와 같은 부서원의 이름, 급여 조회
(단일행 서브쿼리)
select department_id, first_name, salary
from employees
where department_id = (select department_id from employees where first_name='Neena')
(90 부서코드 1개)

- Jennifer와 같은 부서원의 이름, 급여 조회
(다중행 서브쿼리)

```
select department_id, first_name, salary
from employees
where department_id   in   (select department_id from employees where first_name='Jennifer')
                                (10 50   부서코드 2개 )
where department_id in (10, 50)
```

단일행 서브쿼리 – 동일비교 "=", in
다중행 서브쿼리 - in (.....)

- Neena의 급여보다 더 많은 급여 받는 사원의 급여와 이름 조회

```
select salary from employees where first_name='Neena';

select first_name, salary from employees
where salary > (select salary from employees where first_name='Neena')
```

- Jennifer 의 급여보다 더 많은 급여 받는 사원의 급여와 이름 조회

```
select salary from employees where first_name='Jennifer';

select first_name, salary from employees
where salary >any (select salary from employees where first_name='Jennifer')

select first_name, salary from employees
where salary >all (select salary from employees where first_name='Jennifer')
```

단일행 서브쿼리 – 동일비교 "=", in
다중행 서브쿼리 - 동일비교 in (.....) , =any (....)

단일행 서브쿼리 – 대소비교 > >= < <=
다중행 서브쿼리 - 대소비교
> any (3600, 4400) ---> 서브쿼리 결과중 어느 1개보다 크다(=최소값보다 크다)
> all (3600, 4400) ---> 서브쿼리 결과 모두보다 크다(최대값보다 크다)
< any   < all

join 2개 테이블 이상 조회

subquery 2단계 select 조회
any in all

6장 select
....

7.1절 데이터타입 / 함수
7.2절 join / subquery