MYONGJI UNIVERSITY

SQL #1 – SQL Query 1

기본 SELECT

SINGLE-ROW Function

MYONGJI UNIVERSITY

기본 SELECT

SELECT

- 데이터베이스에서 원하는 데이터를 검색, 추출
- Syntax
  - **SELECT** [ALL | DISTINCT] 열_리스트
    [**FROM** 테이블_리스트]
    [**WHERE** 조건]
    [**GROUP BY** 열_리스트 [HAVING 조건]]
    [**ORDER BY** 열_리스트 [ASC | DESC]];
- 기능
  - Projection: 원하는 컬럼 선택
  - Selection: 원하는 튜플 선택
  - Join: 두개의 테이블 결합
  - 기타: 각종 계산, 정렬, 요약(Aggregation)

SELECT의 기능

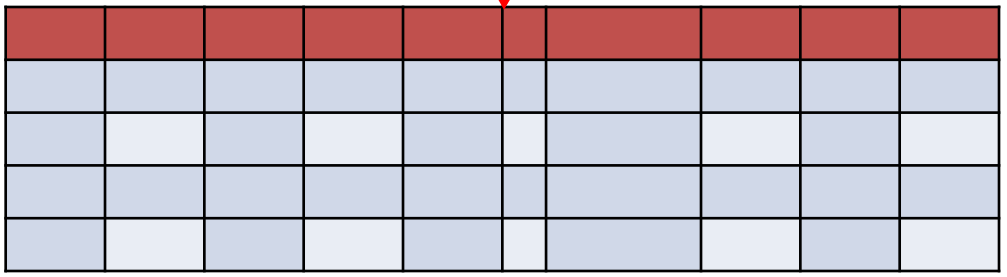
Projection

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	880		20
7499	ALLEN	SALESMAN	7698	81/02/20	1760	300	30
7521	WARD	SALESMAN	7698	81/02/22	1375	500	30
7566	JONES	MANAGER	7839	81/04/02	2975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1375	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7839	KING	PRESIDENT		81/11/17	5000		10
7844	TURNER	SALESMAN	7698	81/09/08	1650	0	30
7876	ADAMS	CLERK	7788	87/05/23	1210		20
7900	JAMES	CLERK	7698	81/12/03	1045		30
7902	FORD	ANALYST	7566	81/12/03	3000		20
7934	MILLER	CLERK	7782	82/01/23	1430		10

Selection

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Join



기본 SELECT

- 형식
  - SELECT *|{[DISTINCT] column|expression [alias],...}
    FROM table;
- 내용
  - * : 모든 컬럼 반환
  - DISTINCT: 중복된 결과 제거
  - SELECT 컬럼명: Projection
  - FROM: 대상 테이블
  - ALIAS: 컬럼 이름 변경 (as)
  - Expression: 기본적인 연산 및 함수 사용 가능

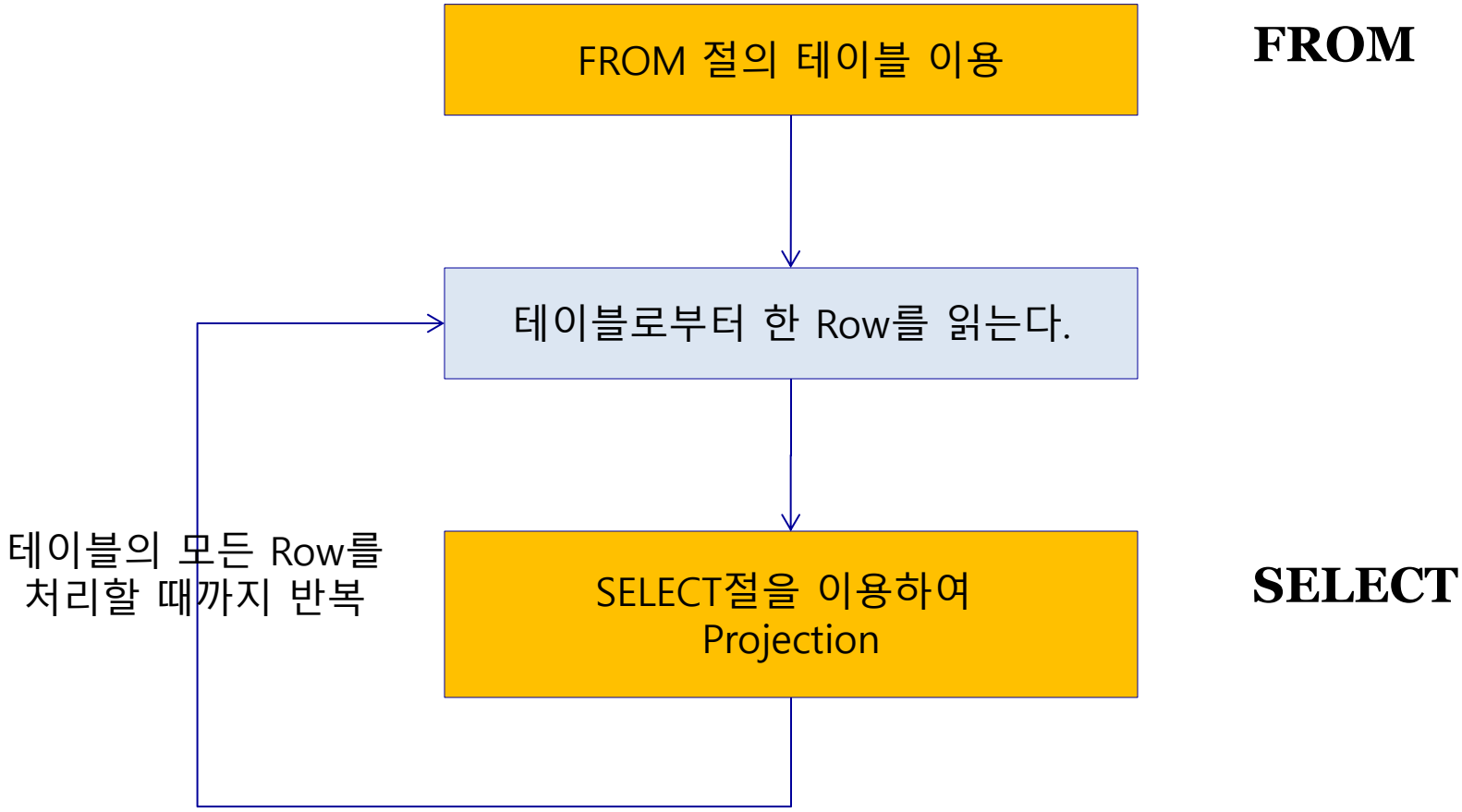
SELECT 실습

- SELECT * FROM emp;
- SELECT ename FROM emp;
- SELECT ename, job FROM emp;
- SELECT ename 이름 FROM emp;

```
SQL> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE          SAL         COMM     DEPTNO
---------- ---------- --------- ---------- -------- ---------- ---------- ----------
      7369 SMITH      CLERK            7902 80/12/17         800                    20
      7499 ALLEN      SALESMAN         7698 81/02/20        1600        300         30
      7521 WARD       SALESMAN         7698 81/02/22        1250        500         30
      7566 JONES      MANAGER          7839 81/04/02        2975                    20
      7654 MARTIN     SALESMAN         7698 81/09/28        1250       1400         30
      7698 BLAKE      MANAGER          7839 81/05/01        2850                    30
      7782 CLARK      MANAGER          7839 81/06/09        2450                    10
      7788 SCOTT      ANALYST          7566 87/04/19        3000                    20
      7839 KING       PRESIDENT             81/11/17        5000                    10
      7844 TURNER     SALESMAN         7698 81/09/08        1500          0         30
      7876 ADAMS      CLERK            7788 87/05/23        1100                    20
      7900 JAMES      CLERK            7698 81/12/03         950                    30
      7902 FORD       ANALYST          7566 81/12/03        3000                    20
      7934 MILLER     CLERK            7782 82/01/23        1300                    10
```

SELECT, FROM 절 처리방법



실제 모든 SQL이 이렇게 처리되는 것은 아닙니다. SQL의 처리 순서는 DBMS가 질의 최적화 과정을 통하여 결정합니다.
질의의 종류, 데이터의 분포 등에 따라 질의의 실제 순서는 달라질 수도 있습니다.

산술연산

- 기본적인 산술연산 사용 가능
  - +, -, *, /, 부호, 괄호 등
  - 우선순위: 부호, * / , + -
  - 컬럼 이름, 숫자
  - 예
    - SELECT ename, (sal+200) * 12 FROM emp;
    - SELECT ename, -sal * 10 FROM emp;

```
SQL> SELECT ename, (sal+200) * 12 FROM emp;

ENAME      (SAL+200)*12
---------- ------------
SMITH             12000
ALLEN             21600
WARD              17400
JONES             38100
MARTIN            17400
BLAKE             36600
```

NULL

- 아무런 값도 정해지지 않았음을 의미
- 어떠한 데이터타입에도 사용가능
- NOT NULL이나 Primary Key 속성에는 사용할 수 없음
- NULL을 포함한 산술식은 일반적으로 NULL
  - SELECT sal, comm, (sal+comm)*12 FROM emp;

- NVL(expr1, expr2)
  - expr1이 NULL이면 expr2를 출력한다.
  - 데이터타입이 호환 가능 하여야 함.
  - SELECT sal, comm, (sal+NVL(comm,0))*12 FROM emp;

Column Alias

- 컬럼의 제목을 변경
- 큰따옴표(" ")을 사용하여 alias내에 공백이나 특수문자를 포함할 수 있다.
- 형태
  - SELECT ename name FROM emp;
  - SELECT ename as name FROM emp;
  - SELECT ename "as" FROM emp;
  - SELECT (sal + comm) "Annual Salary" FROM emp;

```
SQL> select empno no, ename as name, job "to do" from emp;

        NO NAME       to do
---------- ---------- ---------
      7369 SMITH      CLERK
      7499 ALLEN      SALESMAN
      7521 WARD       SALESMAN
      7566 JONES      MANAGER
      7654 MARTIN     SALESMAN
      7698 BLAKE      MANAGER
      7782 CLARK      MANAGER
      7788 SCOTT      ANALYST
      7839 KING       PRESIDENT
      7844 TURNER     SALESMAN
      7876 ADAMS      CLERK
```

Literal

- SELECT 절에 사용되는 문자, 숫자, Date 타입 등의 상수
- Date 타입이나 문자열은 작은따옴표 (' ')로 둘러싸야 함
- 문자열 결합(Concatunation) 연산자: ||
- 예
  - SELECT ename, 1000, SYSDATE FROM emp;
  - SELECT 'Name is ' || ename || ' and no is ' || empno FROM emp;

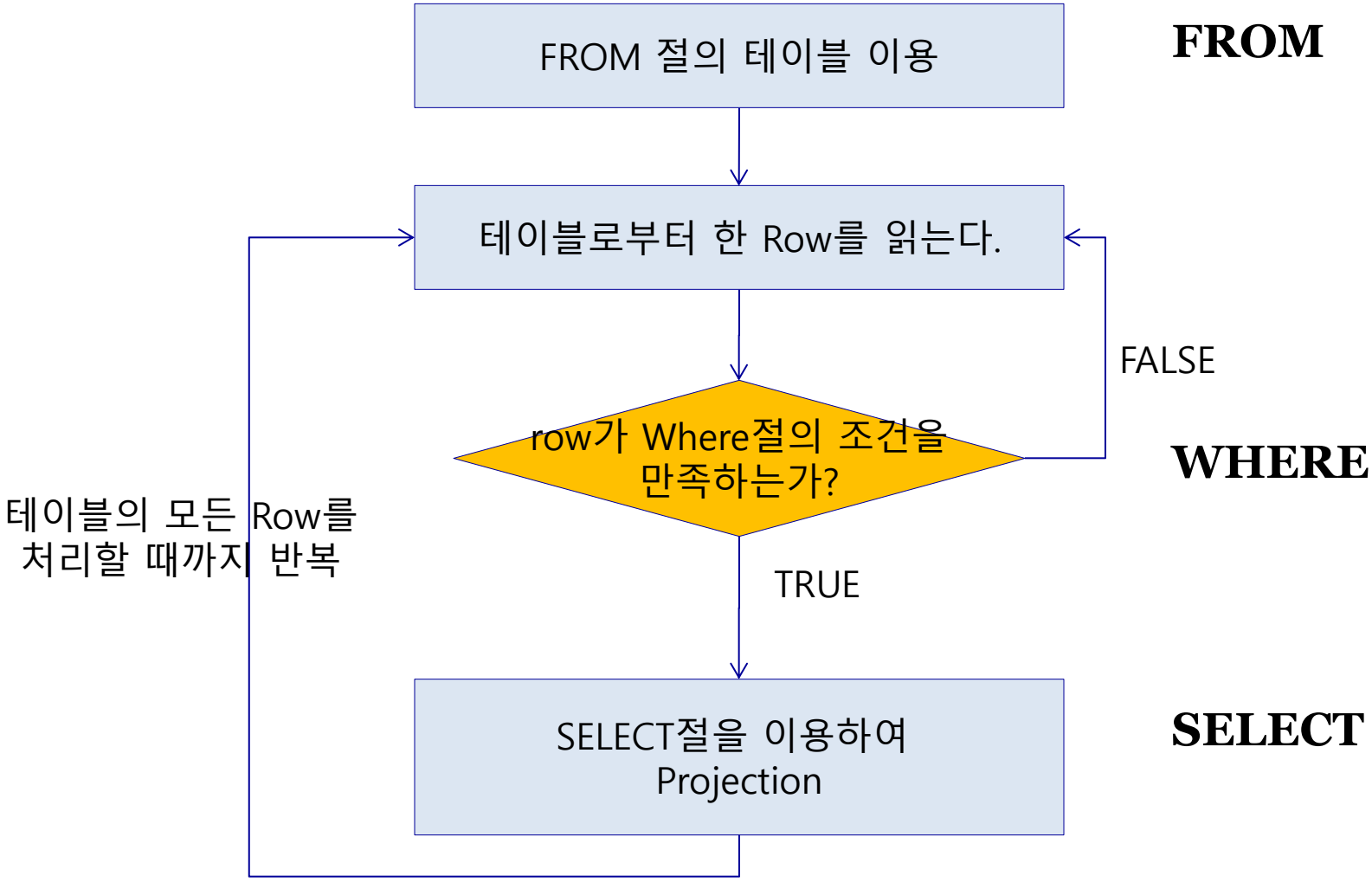
```
SQL> SELECT 'Name is ' || ename || ' and no is ' || empno FROM emp;

'NAMEIS'||ENAME||'ANDNOIS'||EMPNO
--------------------------------------------------------------------------------
Name is SMITH and no is 7369
Name is ALLEN and no is 7499
Name is WARD and no is 7521
Name is JONES and no is 7566
Name is MARTIN and no is 7654
Name is BLAKE and no is 7698
Name is CLARK and no is 7782
Name is SCOTT and no is 7788
```

WHERE

- 조건을 부여하여 만족하는 ROW Selection
- 연산자
  - =, !=, >, <, <=, >=
  - IN : 집합에 포함되는가?
  - BETWEEN a AND b : a 와 b 사이?
  - LIKE: 문자열 부분 검색
  - IS NULL, IS NOT NULL: NULL인지 검색
  - AND, OR: 둘다 만족? 둘 중 하나만 만족?
  - NOT: 만족하지 않음?
  - ANY, ALL : 집합 중 어느한열, 집합 중 모든 열 (다른 비교연산자와 함께 사용)
  - EXIST: 결과 Row가 하나라도 있나? (subquery에서)

WHERE 절 처리 방법



실제 모든 SQL이 이렇게 처리되는 것은 아닙니다. SQL의 처리 순서는 DBMS가 질의 최적화 과정을 통하여 결정합니다.
질의의 종류, 데이터의 분포 등에 따라 질의의 실제 순서는 달라질 수도 있습니다.

LIKE연산

- Wildcard를 이용한 문자열 부분 매칭
- Wildcard
  - % : 임의의 길이의 문자열 (공백 문자 가능)
  - _ : 한 글자
- Escape
  - **ESCAPE** 뒤의 문자열로 시작하는 문자는 Wildcard가 아닌 것으로 해석
- 예
  - ename **LIKE 'KOR%'** : 'KOR'로 시작하는 모든 문자열(KOR가능)
  - ename **LIKE 'KOR_'** : 'KOR'다음에 하나의 문자가 오는 모든 문자열
  - ename **LIKE 'KOR/%%' ESCAPE '/'** : 'KOR%'로 시작하는 모든 문자열

연산자 우선 순위

① Arithmetic operators
② Concatenation operator
③ Comparison conditions
④ IS[NOT] NULL, LIKE, [NOT] IN
⑤ [NOT] BETWEEN
⑥ NOT logical condition
⑦ AND logical condition
⑧ OR logical condition

논리 연산자의 결과값

- NULL을 주의:
  - NULL이 있으면 기본적으로 NULL, 확실히 답이 나오는 경우만 계산 가능

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

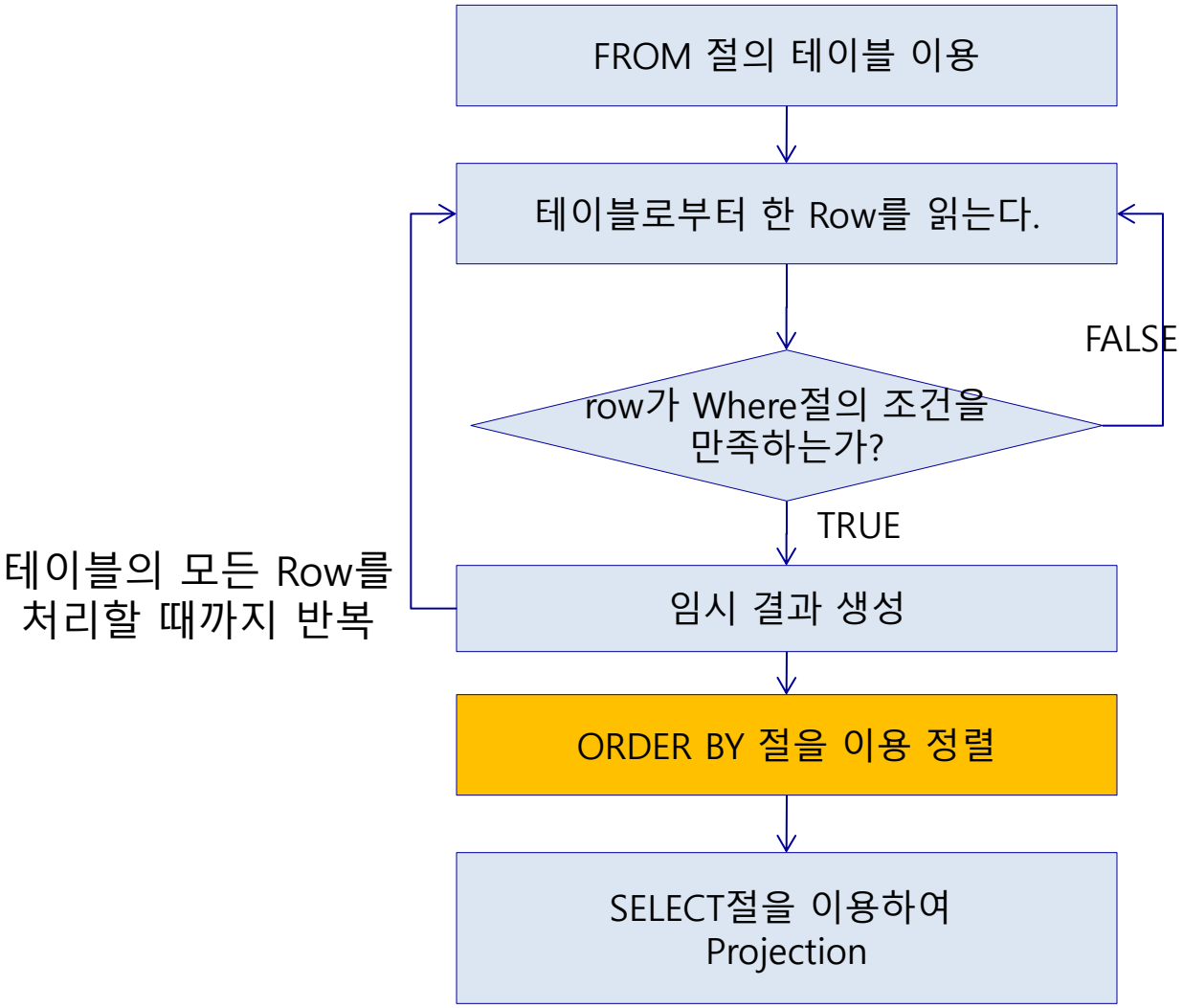
AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

ORDER BY

- 주어진 컬럼 리스트의 순서로 결과를 정렬
- 결과 정렬 방법
  - **ASC** : 오름차순 (작은값→큰값) (default)
  - **DESC**: 내림차순(큰값→작은값)
- 여러 컬럼 정의 가능
  - 첫번째 컬럼이 같으면 두번째 컬럼으로, 두번째 컬럼도 같으면…
- 컬럼 이름대신 Alias, expr, SELECT 절상에서의 순서(1, 2, 3…) 도 사용가능
  - 예) SELECT * FROM emp **ORDER BY** deptno, sal **DESC**
    - 부서번호순으로 정렬하고, sal가 높은 사람부터 출력하시오

ORDER BY절 처리 방법



실제 모든 SQL이 이렇게 처리되는 것은 아닙니다. SQL의 처리 순서는 DBMS가 질의 최적화 과정을 통하여 결정합니다.
질의의 종류, 데이터의 분포 등에 따라 질의의 실제 순서는 달라질 수도 있습니다.

MYONGJI UNIVERSITY

SINGLE ROW FUNCTION

SQL Function

- Single-Row Function : 하나의 Row를 입력으로 받는 함수
  - 숫자함수
  - 문자함수
  - 날짜함수
  - 변환함수
  - 기타함수
- Aggregation Function: 집합함수
- Analytic Function: 분석함수
- Regular Expression: 정규표현식 (Oracle 10g 이상)

문자열 함수

Function	설명
CONCAT(s1,s2)	문자열 결함
INITCAP(s)	첫글자만 대문자로 변경
LOWER(s)	소문자로 변경
UPPER(s)	대문자로 변경
LPAD(s1,n,s2)	문자열의 왼쪽 채움 (길이:n, 채움문자 s2)
RPAD(s1,n,s2)	문자열 오른쪽 채움 (길이:n, 채움문자 s2)
LTRIM(s,c)	문자열 왼쪽 c문자열 제거
RTRIM(s,c)	문자열 오른쪽 c문자열 제거
CHR(n)	ASCII값이 n인 문자 반환
REPLACE(s,p,r)	문자열 치환, S속의 p문자열을 r로 치환
SUBSTR(s,m,n)	부분 문자열, m번째부터 길이 n인 문자열 반환
TRANSLATE(s,from,to)	s에서 from 문자열의 각 문자를 to문자열의 각 문자로 변환
ASCII(s)	ASCII값 반환
INSTR(s1,s2,m,n)	문자열 검색, s1의 m번째부터 s2 문자열이 나타나는 n번째 위치 반환
LENGTH(s)	문자열 길이 반환

문자열 함수 예

- 대소문자 변환

Function	Result
LOWER('Database system')	database system
UPPER('Database system')	DATABASE SYSTEM
INITCAP('Database system')	Database System

- 문자열 조작

함수	결과
CONCAT('Data', 'Base')	DataBase
SUBSTR('Database',2,4)	atab
LENGTH('database')	8
INSTR('Database', 'b')	5
LPAD(salary,10,'*')	*****24000
RPAD(salary, 10, '*')	24000*****
TRIM('#' FROM '##Database###')	Database

숫자 함수

Function	설명	Example	Result
ABS(n)	절대값	ABS(-5)	5
CEIL(n)	n 보다 크거나 같은 최소 정수	CEIL(-2.4)	-2
FLOOR(n)	n 보다 작거나 같은 최대 정수	FLOOR(-2.4)	-3
MOD(m,n)	나머지	MOD(13,2)	1
POWER(m,n)	m의 n승	POWER(2,3)	8
ROUND(m,n)	소수점아래 n자리까지 반올림	ROUND(4.567,2)	4.57
TRUNC(m,n)	소수점아래 n자리미만 버림	TRUNC(4.567,2)	4.56
SIGN(n)	부호 (1, 0, -1)	SIGN(-10)	-1

Date 타입

- Oracle의 Date Type
  - century, year, month, day, hours, minutes, seconds 등 포함한 내부 표현 (7Bytes)
  - Date Format에 따라 출력, 입력됨
  - 기본 Date Format: 'RR/MM/DD' or 'DD-MON-RR'
    - RR은 Y2K고려 2자리 년도 표기 (00~49:2000년대 / 50~99: 1900년대)
  - 포멧 확인
    - SELECT value FROM nls_session_parameters WHERE parameter = 'NLS_DATE_FORMAT';

Date 함수

Function	Purpose
ADD_MONTHS(d,n)	d날짜에 n달 더함
LAST_DAY(d)	d의 달의 마지막 날
MONTHS_BETWEEN(d1,d2)	d1, d2사이의 달 수
NEW_TIME(d,z1,z2)	z1타임존의 d에서 z2타임존의 날짜 생성
NEXT_DAY(d,day)	d날 후의 첫 day요일의 날짜
ROUND(d,fmt)	fmt에 따른 날짜반올림
TRUNC(d,fmt)	fmt에 따른 날짜반올림
SYSDATE	현재 날짜 시간 반환

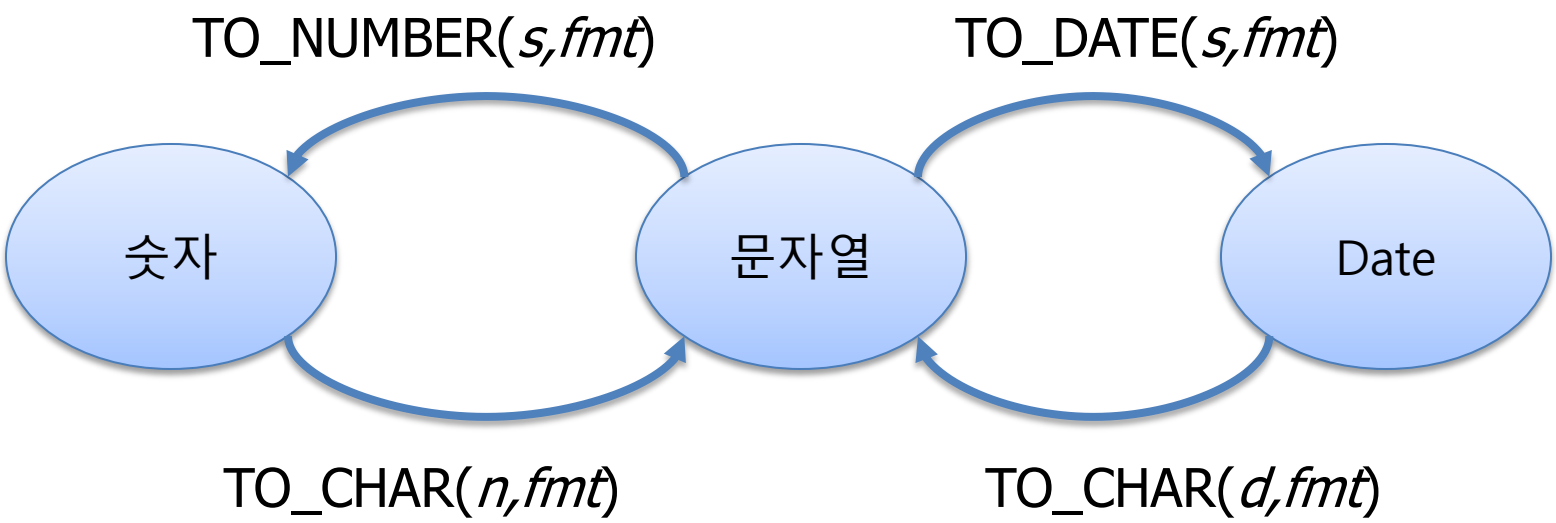
Date함수 예

FUNCTION	RESULT
MONTHS_BETWEEN ('01-SEP-95','11-JAN-94')	19.677419
ADD_MONTHS ('11-JAN-94',6)	'11-JUL-94'
NEXT_DAY ('01-SEP-95','FRIDAY')	'08-SEP-95'
LAST_DAY('01-FEB-95')	'28-FEB-95'

현재날짜를 '25-JUL-95'가정	
ROUND(SYSDATE,'MONTH')	01-AUG-95
ROUND(SYSDATE ,'YEAR')	01-Jan-96
TRUNC(SYSDATE ,'MONTH')	01-JUL-95
TRUNC(SYSDATE ,'YEAR')	01-JAN-95

변환 함수

- 묵시적 변환: 변환함수 없이도 어느정도는 자동으로 변환됨
- 자동으로 변환되지 않을때는 명시적인 변환 함수 사용



Date 변환 포맷

fmt	Description	Example
SS	Second.	0 – 59
SSSSS	Seconds past midnight.	0 – 86399
MI	Minute.	0 – 59
HH, HH24	Hour of day.	0 – 12,23
AM,PM	Meridian indicator.	AM,PM
DD	Day of month.	1 – 31
DAY	Name of day.	SUNDAY – SATURDAY
DY	Abbreviated name of day.	SUN – SAT
D	Day of week.	1 –7
DDD	Day of year.	1 – 366

fmt	Description	Example
W	Week of month.	1 – 5
WW	Week of year.	1 – 53
MM	Two-digit numeric abbreviation of month.	1 – 12
MON	Abbreviated name of month.	JAN – DEC
MONTH	Name of month.	JANUARY – DECEMBER
Q	Quarter of year.	1 – 4
RM	Roman numeral month.	I – XII
AD,BC	AD, BC Indicator.	AD, BC
Y,YY,YYY	1,2,3-digit year.	
YYYY, SYYYY	4-digit year. "S" prefixes BC dates with "-".	

Date 변환 포맷 (계속)

fmt	Description	Example
YEAR, SYEAR	Year, spelled out. "S" prefixes BC dates with "-".	
RR	Given a year with 2 digits. Returns a year in the next century if the year is <50 and the last 2 digits of the current year are >=50. Returns a year in the preceding century if the year is >=50 and the last 2 digits of the current year are <50.	
RRRR	Round year.	
CC, SCC	One greater than the first two digits of a four-digit year; "S" prefixes BC dates with "-".	
J	Julian day. the number of days since January 1, 4712 BC.	
SP	Spelled number.	
TH	Ordinal number.	

숫자 변환 포맷

fmt	Description	Example
9	숫자	99999
0	강제로 0 출력	09999
,	지정된 위치에 ,	99,999
.	소수점	999.99
$	$ 마크	$99999
FM	앞부분의 채움 문자(공백) 없앰.	FM90.9
L	Local 화폐단위	L99,999
MI	음수에 – 부호	99999MI
PR	음수에 괄호	99999PR
RN	로마자 (대소문자 따라 다름)	RN rn
S	부호 기호	S99,999
X	16진수	XXX xxx

기타 함수

- NULL 관련
  - **NVL** (*expr1, expr2*): expr1이 NULL이면 expr2, 아니면 expr1
  - **NVL2** (*expr1, expr2, expr3*): expr1이 NOT NULL이면 expr2, 아니면 expr3
  - **NULLIF** (*expr1, expr2*): 두 식이 같으면 NULL 아니면 expr1
  - **COALESCE** (*expr1, expr2,...exprN*): 첫 NOT NULL인 식, 없으면 expN
- 데이터 타입에 주의
- 예
  - SELECT ename, NVL(TO_CHAR(mgr), 'No Manager')
    FROM emp

Condition Expression

- CASE
  - SELECT ename, job, sal, **CASE** job **WHEN** 'CLERK' **THEN** 1.10*sal
    WHEN 'MANAGER' **THEN** 1.15*sal
    WHEN 'PRESIDENT' **THEN** 1.20*sal
    ELSE sal **END** REVISED_SALARY
    FROM emp;

- DECODE
  - SELECT ename, job, sal, **DECODE**(job, 'CLERK', 1.10*sal,
    'MANAGER', 1.15*sal,
    'PRESIDENT', 1.20*sal,
    sal) REVISED_SALARY
    FROM emp;

학습 마무리

- SELECT 문 (1)
  - SELECT 절: Projection
  - FROM 절: 대상 테이블
  - WHERE 절: Selection 조건
  - ORDER BY 절: 정렬 조건

- 단일행 함수
  - 숫자, 문자열, 날짜 용 함수
  - 변환 함수 : 숫자<-> 문자열, 날짜 <-> 문자열
  - 기타 함수
  - Reference 참조