**1 SQL\*Plus 명령어**

SQL\*Plus 명령어는 오라클 서버(데이터베이스)와 연관되어 작용하는 명령어는 아니며 사용자가 보다 효율적으로 SQL을 활용할 수 있도록 환경을 제공하는 TOOL이다.

**1.1 SQL과 SQL\*Plus 의 차이점**

**1.1.1 SQL의 특징**

1. RDBMS의 표준 언어
2. SQL Buffer에 바로 전에 실행한 SQL문장이 저장
3. 명령의 끝은 ;을 사용한다

**가) 명령어의 종류**

|  |  |
| --- | --- |
| 구 분 | 종 류 |
| Data Retrieval Language | SELECT |
| Data Manipulation Language | INSERT, UPDATE, DELETE, COMMIT, ROLLBACK |
| Data Definition Language | CREATE, ALTER, DROP, RENAME |
| Data Control Language | GRANT, REVOKE, AUDIT, NO AUDIT, LOCK |
| Miscellaneous Language | COMMENT |

**1.1.2 SQL\*Plus의 특징**

1. SQL 문장을 실행 한다.
2. SQL Buffer에 저장되지 않는다.
3. SQL 문장을 편집한다.
4. 환경 및 기본 조회 결과를 FORMATTING

**가) 명령어의 종류**

|  |  |
| --- | --- |
| 구 분 | 설 명 |
| Execution Commands | /, RUN, EXECUTE |
| Edit Commands | LIST, APPEND, CHANGE, DEL, INPUT, EDIT |
| Environment Commands | SET, SHOW, PAUSE |
| Report Format Commands | COLUMN, CLEAR, BREAK, COMPUTE, TTITLE, BTITLE |
| File Manipulation Commands | SAVE, GET, START, @, @@, SPOOL |
| Interactive Commands | DEFINE, UNDELETE, PROMPT, ACCEPT, VARIABLE, PRINT |
| Database Access Commands | CONNECT, COPY, DISCONNECT |
| Miscellaneous Commands | SQLPLUS,EXIT,HELP,DESCRIBE,HOST,REMARK,WHENEVER |

**1.2 SQL명령 편집 및 실행**

**1.2.1 SQL Buffer에는 SQL명령어의 편집**

SQL Buffer에 있는 명령어를Line단위로 편집할 수 있다.

**가) 종류**

|  |  |
| --- | --- |
| 명 령 어 | 설 명 |
| A[PPEND] text | 현재 편집 라인의 끝에 text를 추가 |
| C[HANGE]/old/new | 현재 편집 라인의 old문자를 new문자로 바꿈 |
| DEL [n] | n 라인을 삭제 |
| I[NPUT] [text] | 현재 편집 라인 다음에 라인을 추가하여 text를 추가 |
| L[IST] [n] | SQL문장을 보여주고, 편집 라인을 이동 |
| n text | n번째 라인을 text로 바꿈 |
| R[UN] | Buffer에 있는 명령어를 실행한다.( /와 동일 ) |

**나) 사용 예**

|  |
| --- |
| SQL> **SELECT empno,ename,jb,dept**  2 **FROM emp;**  SELECT empno,ename,jb,dept  \*  ERROR at line 1:  ORA-00904: invalid column name  SQL> **c/jb/job**  1\* SELECT empno,ename,job,dept  SQL> **a no**  1\* SELECT empno,ename,job,deptno  SQL> **/**  EMPNO ENAME JOB DEPTNO  --------- ---------- --------- ---------  7839 KING PRESIDENT 10  7698 BLAKE MANAGER 30  7782 CLARK MANAGER 10  7566 JONES MANAGER 20  7654 MARTIN SALESMAN 30  7499 ALLEN SALESMAN 30  7844 TURNER SALESMAN 30  7900 JAMES CLERK 30  7521 WARD SALESMAN 30  7902 FORD ANALYST 20  7369 SMITH CLERK 20  7788 SCOTT ANALYST 20  . . . . . . . . . .  14 rows selected. |

**1.2.2 파일에 있는 명령어 편집과 실행**

SCRIPT를 생성하거나 LINE단위의 편집이 아니라 SCREEN편집을 할 경우에 유용하다. SQL\*Plus 파일 명령어는 파일을 저장, 획득, 적재 그리고 실행하는데 사용할 수 있다.

**가) 종류**

|  |  |
| --- | --- |
| 명 령 어 | 설 명 |
| EDIT [filename[.ext]] | 지정된 파일의 내용이나 버퍼의 내용을 운영체제의 문자 편집기로 불러온다. |
| SAV[E] [filename[.ext]]  [REP[LACE] | APP[END]] | SQL 버퍼의 내용을 파일에 저장하고 기존 파일에 추가하기 위해서 APPEND를, 기존 파일에 중복해서 쓰려면 REPLACE를 사용한다. 기본적인 파일 확장자는 sql이다. |
| STA[RT] [filename[.ext]] | 지정된 파일을 수행한다. START라는 명령 대신에 @을 사용할 수 있다. 파일 확장자가 .sql이 아니면, 파일 확장자를 써야 한다. |
| GET [filename[.ext]] | SQL 버퍼에 파일의 내용을 기록한다. 파일명의 기본적인 확장자는 .lis또는 .lst이다. |
| SPO[OL] [filename[.ext]]  | OFF | OUT] | SQL\*Plus의 내용(Query 결과)을 파일에 저장한다. OFF는 스풀 파일을 닫는다. |
| HOST | SQL\*Plus안에서 호스트 운영체제의 명령어를 실행한다. |
| ! | UNIX Shell로 나들이 |
| !vi file\_name.sql | file\_name.sql을 vi 편집기로 부름 |

♣ 참고

파일명을 쓰지 않고 EDIT명령어를 사용할 때, 기본적인 파일명은 afiedt.buf이다. 이 파일은 버퍼를 편집할 때마다 이 파일에 겹쳐 쓰게 된다. 현재의 디렉토리에 없는 파일명을 명시하면 SQL\*Plus는 파일 이름을 묻는다.

**나) 사용 예**

|  |
| --- |
| SQL> **SELECT e.empno,e.ename,e.job,e.sal,d.dname,d.loc**  2 **FROM dept d,emp e**  3 **WHERE d.deptno = e.deptno;**  EMPNO ENAME JOB SAL DNAME LOC  --------- ---------- --------- --------- -------------- -------------  7839 KING PRESIDENT 5000 ACCOUNTING NEW YORK  7698 BLAKE MANAGER 2850 SALES CHICAGO  7782 CLARK MANAGER 2450 ACCOUNTING NEW YORK  7566 JONES MANAGER 2975 RESEARCH DALLAS  7654 MARTIN SALESMAN 1250 SALES CHICAGO  7499 ALLEN SALESMAN 1600 SALES CHICAGO  7844 TURNER SALESMAN 1500 SALES CHICAGO  7900 JAMES CLERK 2450 SALES CHICAGO  . . . . . . . . . .  14 rows selected.  SQL> **save emp\_dept**  Created file emp\_dept  SQL> **ed emp\_dept**  File의 마지막 부분에 “and e.job = 'MANAGER'”를 추가 후 저장하고 종료한다.  SELECT e.empno,e.ename,e.job,e.sal,d.dname,d.loc  FROM dept d,emp e  WHERE d.deptno = e.deptno  AND e.job = 'MANAGER'  /  SQL> **@emp\_dept**  EMPNO ENAME JOB SAL DNAME LOC  --------- ---------- --------- --------- -------------- --------  7698 BLAKE MANAGER 2850 SALES CHICAGO  7782 CLARK MANAGER 2450 ACCOUNTING NEW YORK  7566 JONES MANAGER 2975 RESEARCH DALLAS |

**1.2.3 SQL\*Plus의 환경 설정**

SQL\*Plus를 사용한 환경 설정은 SET명령을 이용한다.

**가) Syntax**

|  |
| --- |
| SET 시스템변수 값 |

시스템변수 세션 환경을 제어하는 변수

값 시스템 변수의 값

**나) 종류**

|  |  |
| --- | --- |
| SET 변수와 값 | 설 명 |
| ARRAY[SIZE] {20|n} | 데이터베이스 데이터 패치의 크기를 설정 |
| COLSEP { |text} | 열 사이에 출력되는 문자를 설정(Default:space) |
| FEED[BACK] {6|n|OFF|ON} | 질의가 최소한 n개이어야 ROW의 수를 출력 |
| HEA[DING] {OFF|ON} | 열의 HEADING을 출력할 지의 여부를 결정 |
| LIN[ESIZE] {80|n} | LINE당 문자의 수를 결정 |
| LONG {80|n} | LONG 값을 출력하기 위해 최대 폭을 설정 |
| PAGES[IZE] {24|n} | PAGE당 LINE수를 지정 |
| PAU[SE] {OFF|ON|text} | 화면 제어를 함 |
| TERM[OUT] {OFF|ON} | 결과를 화면에 출력할 지의 여부를 결정 |

☞ Guidelines

1. SET명령어는 현재의 세션(운영 중인 SQL\*Plus) 환경을 제어
2. login.sql에는 로그온시 실행되는 표준 SET 명령과 그 외의 SQL\*Plus명령을 기술
3. login.sql을 수정하여 부가적인 SET 명령을 사용

♣ 참고

login.sql 파일에는 접속할 때마다 필요한 표준 SET 명령과 그 외의 SQL\*Plus 명령어들이 들어 있다. 접속할 때에 이 파일을 읽어서 명령어가 수행된다. 로그 아웃을 하면 사용자 정의 설정이 상실된다. 설정 값을 영구적으로 변경하려면 login.sql 파일에 추가한다.

**다) 사용 예**

|  |
| --- |
| SQL> **SET HEADING OFF**  SQL> **SET COLSEP '\*\*'**  SQL> **SELECT \***  2 **FROM ITEM;**  610\*\* 3\*\* 100890\*\* 58\*\* 1\*\* 58  611\*\* 1\*\* 100861\*\* 45\*\* 1\*\* 45  612\*\* 1\*\* 100860\*\* 30\*\* 100\*\* 3000  601\*\* 1\*\* 200376\*\* 2.4\*\* 1\*\* 2.4  602\*\* 1\*\* 100870\*\* 2.8\*\* 20\*\* 56  604\*\* 1\*\* 100890\*\* 58\*\* 3\*\* 174  604\*\* 2\*\* 100861\*\* 42\*\* 2\*\* 84  604\*\* 3\*\* 100860\*\* 44\*\* 10\*\* 440  603\*\* 2\*\* 100860\*\* 56\*\* 4\*\* 224  . . . . . . . . .  64 rows selected. |

**1.2.4 SQL\*Plus FORMAT 명령어**

다음의 명령어를 사용하여 리포트의 속성을 제어할 수 있다.

|  |  |
| --- | --- |
| 명 령 어 | 설 명 |
| COL[UMN] [column option] | 열 포맷을 제어한다. |
| TTI[TLE] [text|OFF|ON] | 리포트의 머리말을 명시한다. |
| BTI[TLE] [text|OFF|ON] | 리포트의 꼬리말을 명시한다. |
| BRE[AK] [ON report\_element] | 중복값을 제거하고 라인 피드로 행들을 단락 짓는다. |

☞ Guidelines

1. 모든 포맷 명령은 SQL\*Plus SESSION의 마지막이나 변경 전까지 효력을 유지한다.
2. 모든 리포트 후에는 SQL\*Plus Default값으로 RESET하는 것을 원칙으로 한다.
3. SQL\*Plus 변수 설정을 Default값으로 해주는 명령은 없다. 특정 값을 알거나 로그 아웃한 후 로그인을 해야 한다.
4. 열에 별칭을 사용하였다면 열 이름이 아닌 별칭을 사용하여야 한다.

**1.2.5 COLUMN 명령어**

열의 출력을 제어합니다.

**가) Syntax**

|  |
| --- |
| COL[UMN] [{column | alias} [option]] |

**나) COLUMN 명령의 OPTION**

|  |  |
| --- | --- |
| OPTION | 설 명 |
| CLE[AR] | 어떤 열의 형식을 해제합니다. |
| FOR[MAT] format | 열 데이터의 디스플레이를 변경합니다. |
| HEA[DING] text | 열 헤딩을 설정합니다.수직 바( )는 헤딩 라인을 나눕니다. |
| JUS[TIFY] [align] | 열 HEADING을 정렬(왼쪽,가운데,오른쪽)합니다. |
| NOPRI[NT] | 열을 숨깁니다. |
| NUL[L] text | Null값에 대해 디스플레이 되는 텍스트를 명시합니다. |
| PRI[NT] | 열을 보여줍니다. |
| TRU[NCATED] | 디스플레이 되는 첫번째 라인의 마지막 문자열을 절삭합니다. |
| WRA[PPED] | 문자열이 끝나면 다음 라인으로 이동합니다. |

**다) 사용 예**

|  |
| --- |
| SQL> **COL ename HEADING '사원의|이름' FORMAT a15**  SQL> **COL ename HEA '사원의|이름' FOR a15**  SQL> **COL sal JUS LEFT FOR $999,990.00**  SQL> **COL mgr FOR 9999999999 NUL 'No manager'**  SQL> **SELECT empno, ename, mgr, sal, deptno**  2 **FROM emp**  3 **WHERE deptno = 10;**  사원의  EMPNO 이름 MGR SAL DEPTNO  --------- --------------- ----------- ------------ ---------  7839 KING No manager $5,000.00 10  7782 CLARK 7839 $2,450.00 10  7934 MILLER 7782 $1,500.00 10  SQL> **COL ename**  column ename ON  heading '사원의|이름' headsep '|'  format a15  SQL> **col ename CLE**  SQL> **COL ename**  column 'ename' not defined |

♣ 참고

SQL\*Plus명령이 너무 길다면 현재 라인의 마지막에 하이폰(\_)을 두어서 다음 라인에 계속적으로 작성할 수 있습니다. 오라클 서버는 형식 모델에서 제공된 숫자의 자리 수를 초과하게 되면 자리 전체에 파운드 기호(#)의 문자열을 출력합니다. 또한 포맷 모델이 알파벳이지만 실제 값은 숫자인 값의 자리에도 파운드 기호(#)를 출력합니다.

**1.3 SQL\*Plus를 이용하여 보고서 작성**

SQL\*Plus 명령어를 이용하여 간단한 보고서를 작성할 수 있다.

|  |
| --- |
| SQL> **SET PAGESIZE 42**  SQL> **SET LINESIZE 54**  SQL> **SET FEEDBACK OFF**  SQL> **TTITLE '사원의 업무별|보고서 양식'**  SQL> **BTITLE 'GOOD BY'**  SQL> **COLUMN job HEADING 'Job|Category' FORMAT A20**  SQL> **COLUMN ename HEADING 'Employee' FORMAT A20**  SQL> **COLUMN sal HEADING 'Salary' FORMAT $999,990.00**  SQL> **BREAK ON job SKIP 1 ON REPORT**  SQL> **COMPUTE SUM OF sal ON job REPORT**  SQL> **SPOOL salary**  SQL> **SELECT job, ename, sal**  2 **FROM emp**  3 **ORDER BY job,ename,sal DESC;** |
| Fri Feb 12 page 1  Suppress Duplicate  Values  Display a Grand Total  Display a Header,  Date and Page Number  사원의 업무별  보고서 양식    Job  Format Column  Headings  Category Employee Salary  -------------------- -------------------- ------------  ANALYST FORD $3,000.00  SCOTT $3,000.00  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ------------  Display Subtotal  sum $6,000.00    CLERK ADAMS $1,100.00  Format Data  JAMES $950.00  MILLER $1,500.00  SMITH $800.00  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ------------  Skip Lines Between  Groups  sum $4,350.00    MANAGER BLAKE $2,850.00  CLARK $2,450.00  JONES $2,975.00  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ------------  sum $8,275.00    PRESIDENT KING $5,000.00  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ------------  sum $5,000.00    SALESMAN ALLEN $1,600.00  MARTIN $1,250.00  TURNER $1,500.00  WARD $1,250.00  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ------------  sum $5,600.00    ------------  sum $29,225.00    Display a Footer  GOOD BY |

**1.3.1 BREAK명령**

행 단락을 구분 짓고 중복 값을 제거하기 위해서 BREAK 명령을 사용합니다. BREAK 명령이 효과적으로 수행되기 위해서 BREAK되는 열에 대해서 ORDER BY하십시오.

**가) Syntax**

|  |
| --- |
| **BREAK ON** olumn[|alias|row] [skip n|dup|page] on . . . . [ON REPORT] |

page break 값이 변경될 때 새로운 PAGE로 SKIP

skip n break 값이 변경될 때 n만큼 줄을 SKIP(COLUMN,ROW,PAGE,REPORT)

duplicate 중복되는 값을 출력

**나) 사용 예**

|  |
| --- |
| SQL> BREAK ON deptno ON job : 중복 제거의 경우  SQL> BREAK ON REPORT : 전체 합계의 경우  SQL> BREAK ON deptno SKIP 2 ON job SKIP 1 : BREAK 값에서 행을 단락짓는 경우 |

**다) CLEAR 명령**

CLEAR 명령은 BREAK POINT설정을 해제한다.

|  |
| --- |
| **CLEAR BREAK** |

**1.3.2 BREAK를 이용한 SELECT문장 기술**

SELECT 문의 ORDER BY 절로 BREAK를 제어한다.

**가) Syntax**

|  |
| --- |
| **SELECT** . . . . .  **FROM** . . . . .  **ORDER BY** break\_column; |

**나) 사용 예**

|  |
| --- |
| SQL> **BREAK ON job SKIP 1 ON REPORT**  SQL> **SELECT job, ename, sal**  2  **FROM emp**  3  **ORDER BY job,ename,sal DESC;** |

☞ Guidelines

1. BREAK 명령을 사용할 때 DATA의 의미 있는 SUBSET를 만들기 위해 BREAK명령에 참조된 COLUMN을 ORDER BY 절에 기술한다.
2. BREAK 명령은 마지막에 기술된 오직 하나의 명령어만 유효하다.

**1.3.4 COMPUTE 명령**

SQL\*Plus 명령어를 이용하여 요약된 계산을 더한다.

**가) Syntax**

|  |
| --- |
| **COMPUTE** function **OF** compute\_column **ON** break\_column |

function COUNT,NUM,MAX,MIN,SUM,AVG,STD,VAR중 하나의 함수를 기술

compute\_column 계산에 사용되는 COLUMN이나 식

break\_column BREAK명령으로 기술된 COLUMN

**나) COMPUTE명령 취소**

현재 설정된 COMPUTE명령을 Clear한다.

|  |
| --- |
| **CLEAR COMPUTE** |

**다) 사용 예**

|  |
| --- |
| SQL> **BREAK ON job SKIP 1 ON REPORT**  SQL> **COMPUTE SUM OF sal ON job REPORT** |

**1.3.5 TTITLE과 BTITLE 명령**

머리말과 꼬리말을 출력합니다.

**가) Syntax**

|  |
| --- |
| **TTI**[**TLE**] [text|**OFF**|**ON**] |

**나) 사용 예**

|  |
| --- |
| SQL> **TTITLE '사원의 업무별|보고서 양식'**  SQL> **BTITLE 'GOOD BY'** |

**1.3.6 REPORT를 실행 하기 위한 SCRIPT FILE 작성**

SQL프롬프트에서 각각의 SQL\*Plus명령을 입력하거나 명령(또는 스크립트)파일SELECT문장을 포함하는 모든 명령어를 입력할 수 있습니다. 전형적인 스크립트는 최소한 하나의 SQL문장과 여러 개의 SQL\*Plus명령어들로 구성되어 있습니다.

**가) 작성 절차**

|  |
| --- |
| SQL> **ed rep**  “ORACLE\_HOME\BIN” DIRECTORY에 rep.sql FILE이 존재하지 않으면 생성 여부를 요구하고 rep.sql FILE이 있으면 FILE을 OPEN한다. OPEN된 FILE에 내용을 입력한 후 저장하고 종료한다.  SQL> **@rep**  위 명령은 SQL> START rep와 동일한 명령으로 SCRIPT의 내용을 실행한다. |

♣ 참고

SCRIPT 작성 중 SQL\*Plus Window를 클릭하여 사용할 수 없다. 반드시 편집중인 Window를 종료 후 SQL\*Plus 사용할 수 있다.

문제1) 아래의 SCRIPT를 분석하여라

|  |
| --- |
| SET PAGESIZE 42  SET LINESIZE 54  SET FEEDBACK OFF  TTITLE '사원의 업무별|보고서 양식'  BTITLE 'GOOD BY'  COLUMN job HEADING 'Job|Category' FORMAT A20  COLUMN ename HEADING 'Employee' FORMAT A20  COLUMN sal HEADING 'Salary' FORMAT $999,990.00  BREAK ON job SKIP 1 ON REPORT  COMPUTE SUM OF sal ON job REPORT  SPOOL salary  SELECT job, ename, sal  FROM emp  ORDER BY job,ename,sal DESC  / |

문제2) 아래의 SCRIPT를 분석하여라

|  |
| --- |
| SET PAGESIZE 36  SET LINESIZE 64  SET FEEDBACK OFF  TTITLE '사원의 업무별|보고서 양식'  BTITLE '수고 하였습니다'  COLUMN job HEADING '담당업무' FORMAT A20  COLUMN ename HEADING '성 명' FORMAT A20  COLUMN sal HEADING '급 여' FORMAT $999,990.00  BREAK ON deptno SKIP 2 ON job SKIP 1 ON REPORT  COMPUTE SUM OF sal ON deptno REPORT  SPOOL emp\_sal  SELECT deptno,job, ename, sal  FROM emp  ORDER BY deptno,job,ename,sal DESC;  SPOOL OFF  CREATE COMPUTE  CREATE BREAK  COLUMN job CLEAR  COLUMN ename CLEAR  COLUMN sal CLEAR  TTITLE OFF  BTITLE OFF  SET PAGESIZE 14  SET LINESIZE 80  SET FEEDBACK ON |

**1.4 상호작용 리포트**

SQL\*Plus를 사용하여 리턴된 자료의 범위를 제한하는데 사용자가 입력하는 값을 이용하기 위해서 사용자에게 PROPMT라는 리포트를 생성할 수 있다. 상호 작용 리포트를 생성하기 위해 명령어 파일이나 단일 SQL문장에 치환 변수를 내장할 수 있다.

**1.4.1 치환 변수**

1. 값을 임시로 저장하기 위해서 SQL\*Plus 치환 변수를 사용합니다.
2. Single ampersand( & )
3. Double ampersand( && )
4. DEFINE과 ACCEPT명령어
5. SQL 문장간에 변수 값을 전달 합니다.
6. 머리말과 꼬리말을 동적으로 변경합니다.

♣ 참고

SQL\*Plus는 사용자 입력에 대한 타당성 검사를 하지 않는다, 사용자에 대해서 만드는 PROMPT는 단순하고 모호하지 않게 하십시오.

**1.4.2 치환 변수 사용할 수 있는 절**

1. WHERE절
2. ORDER BY
3. COLUMN 표현식
4. 테이블 이름
5. 전체 SELECT 문장

**1.4.3 Single ampersand( & )의 치환 변수**

리포트에서 실행할 때 사용자는 종종 리턴되는 데이터를 동적으로 제한 하기를 원한다. SQL\*Plus는 사용자 변수로써 이러한 융통성을 제공합니다. SQL에서 각각의 변수를 인식하기 위해서 “&”를 사용합니다. 숫자는 &dept\_no와 같이 인용 부호를 사용하지 않고 사용하고 문자와 날짜에 대해서는 단일 인용 부호(‘&name’)를 사용하면 보다 편리하게 사용할 수 있다.

문제3) EMP 테이블에서 부서번호를 입력받아 동적 조회할 수 있는 SELECT문장을 기술하여라.

|  |
| --- |
| SQL> **SELECT empno,ename,job,hiredate,deptno**  2 **FROM emp**  3 **WHERE deptno = &dept\_no;**  Enter value for dept\_no: **10**  old 3: WHERE deptno = &dept\_no  new 3: WHERE deptno = 10  EMPNO ENAME JOB HIREDATE DEPTNO  --------- ---------- --------- ------------------ ---------  7839 KING PRESIDENT 17-NOV-81 10  7782 CLARK MANAGER 09-JUN-81 10  7934 MILLER CLERK 23-JAN-82 10 |

참고) SET VERIFY ON/OFF명령어를 이용하여 old와 new의 출력을 조절할 수 있다.

|  |
| --- |
| SQL> **SET VERIFY OFF**  SQL> **SELECT empno,ename,job,hiredate,deptno**  2 **FROM emp**  3 **WHERE deptno = &dept\_no;**  Enter value for dept\_no: **10**  EMPNO ENAME JOB HIREDATE DEPTNO  --------- ---------- --------- ------------------ ---------  7839 KING PRESIDENT 17-NOV-81 10  7782 CLARK MANAGER 09-JUN-81 10  7934 MILLER CLERK 23-JAN-82 10  SQL> SET VERIFY ON |

문제4) EMP 테이블에서 이름을 입력받아 동적 조회할 수 있는 SELECT문장을 기술하여라.

|  |
| --- |
| SQL> **SELECT empno,ename,job,hiredate,sal,deptno**  2 **FROM emp**  3  **WHERE UPPER(ename) = UPPER('&name');**  Enter value for name: **scott**  old 3: WHERE ename = UPPER('&name')  new 3: WHERE ename = UPPER('scott')  EMPNO ENAME JOB HIREDATE SAL DEPTNO  --------- ---------- --------- ------------------ --------- ---------  7788 SCOTT ANALYST 09-DEC-82 3000 20 |

문제5) EMP 테이블에서 사원번호는 반드시 출력하고 나머지 열은 입력받아 출력하고 조건도 입력받아 출력하여라

|  |
| --- |
| SQL> **SELECT empno,&column\_name**  2 **FROM emp**  3 **WHERE &condition;**  Enter value for column\_name: **ename,job,deptno**  old 1: SELECT empno,&column\_name  new 1: SELECT empno,ename,job,deptno  Enter value for condition: **deptno = 10**  old 3: WHERE &condition  new 3: WHERE deptno = 10  EMPNO ENAME JOB DEPTNO  --------- ---------- --------- ---------  7839 KING PRESIDENT 10  7782 CLARK MANAGER 10  7934 MILLER CLERK 10 |

**1.4.4 Double ampersand( && )의 치환 변수**

매번 사용자에게 입력받지 않고 입력된 변수의 값을 사용하고자 할 경우에 사용한다. 사용자는 오직 한번만 입력하면 된다.

문제6) EMP 테이블에서 사원번호,이름,업무는 반드시 출력하고 나머지 열은 입력받아 출력하고 입력받은 열을 정렬 조건으로 사용한다.

|  |
| --- |
| SQL> **SELECT empno,ename,job,&&column\_name**  2 **FROM emp**  3 **ORDER BY &column\_name;**  Enter value for column\_name: **deptno**  old 1: SELECT empno,ename,job,&&column\_name  new 1: SELECT empno,ename,job,deptno  old 3: ORDER BY &column\_name  new 3: ORDER BY deptno  EMPNO ENAME JOB DEPTNO  --------- ---------- --------- ---------  7839 KING PRESIDENT 10  7782 CLARK MANAGER 10  . . . . . . . . . .  14 rows selected. |

**1.4.5 사용자 변수의 정의**

SELECT 문장을 실행하기 전에 사용자 변수를 미리 정의해 사용할 수 있습니다. SQL\*Plus는 사용자 변수를 정의하고 설정하기 위해 두개의 명령어를 제공합니다.

|  |  |
| --- | --- |
| 명 령 어 | 설 명 |
| DEFINE variable = value | CHAR 데이터형 사용자 변수를 생성하고 값을 할당합니다. |
| DEFINE variable | 변수,변수 값,변수 데이터형을 출력합니다. |
| DEFINE | 값과 데이터형을 가진 모든 데이터형을 출력합니다. |
| ACCEPT | 사용자 입력 라인을 읽고 그것을 변수에 저장합니다 |

**가) ACCEPT 명령어**

1. 사용자 입력을 받을 때 사용자가 원하는 프롬프트를 생성합니다.
2. NUMBER또는 DATE데이터형 변수를 명시적으로 정의합니다.
3. 보안의 이유 때문에 사용자 입력을 숨깁니다.

|  |
| --- |
| **ACCEPT**  variable [datatype] [**FOR**[**MAT**] format] [**PROMPT** text] [**HIDE**] |

variable 값을 저장하는 변수의 이름입니다.

존재하지 않으면 SQL\*Plus가 그것을 생성하여 사용합니다.

Datatype NUMBER,CHAR또는 DATE.CHAR는 최대 길이 제한이240bytes입니다.

DATE는 형식 모델을 다시 검사하고 데이터형은 CHAR입니다.

format 형식 모델을 명시합니다.(예:A10, 9.999)

text 사용자가 값을 입력하기 전에 값을 출력합니다.

HIDE 사용자 입력을 숨긴다.(예:패스워드)

♣ 주의

ACCEPT 명령에서 치환 매개변수를 참조할 때 SQL\*Plus치환 매개변수 앞에 앰퍼샌드(&)를 두어서는 안됩니다.

문제7) ACCEPT명령으로 업무를 입력받아 사원번호,이름,업무,급여를 출력하여라

|  |
| --- |
| SQL> **ACCEPT job PROMPT '담당업무를 입력하시오: '**  담당업무를 입력하시오: **manager**  SQL> **SELECT empno,ename,job,sal**  2 **FROM emp**  3 **WHERE UPPER(job) = UPPER('&job');**  old 3: WHERE UPPER(job) = UPPER('&job')  new 3: WHERE UPPER(job) = UPPER('manager')  EMPNO ENAME JOB SAL  --------- ---------- --------- ---------  7698 BLAKE MANAGER 2850  7782 CLARK MANAGER 2450  7566 JONES MANAGER 2975 |

**나) DEFINE과 UNDEFINE 명령어**

1. 변수는 다음의 경우까지 계속 정의 됩니다.
2. 선언된 변수에 대해 UNDEFINE명령을 사용
3. SQL\*Plus종료
4. DEFINE명령으로 변수 내용을 검사할 수 있습니다.
5. 모든 SESSION에 대해 변수를 정의하기 위해서는 login.sql file을 수정하면 변수는 Startup시 생성되어 사용할 수 있습니다.

문제8) 부서이름(sales) 유지하기 위해 변수(dept\_name)를 선언한 다음 이 변수를 이용하여 DEPT 테이블을 검색하여라

|  |
| --- |
| SQL> **DEFINE dept\_name = sales**  SQL> **DEFINE dept\_name**  DEFINE DEPT\_NAME = "sales" (CHAR)  SQL> **SELECT \***  2 **FROM dept**  3 **WHERE UPPER(dname) = UPPER('&dept\_name');**  old 3: WHERE UPPER(dname) = UPPER('&dept\_name')  new 3: WHERE UPPER(dname) = UPPER('sales')  DEPTNO DNAME LOC  --------- -------------- -------------  30 SALES CHICAGO |

**◈ 연 습 문 제 ◈**

1. 단일 앤퍼샌드(&) 치환 변수를 설명하시오.

2. EMP 테이블에서 이름과 업무는 ,로 구분하여 출력하고 입사일자는 YYYY년MM월DD일 X요일형태로 출력하는 SELECT 문장을 기술하시오.

3. EMP 테이블에서 실행시 이름을 입력받아 이름,업무,급여를 출력하는 SCRIPT를 작성하시오.

|  |
| --- |
| Please enter the location name : Dallas  old 4: AND loc = UPPER('&p\_name')  new 4: AND loc = UPPER('Dallas')  EMPLOYEE JOB DEPT\_NAME  ---------- --------- --------------  JONES MANAGER RESEARCH  FORD ANALYST RESEARCH  SMITH CLERK RESEARCH  SCOTT ANALYST RESEARCH  ADAMS CLERK RESEARCH |

4. EMP 테이블에서 입력되는 지역에 대해서 부서명,이름,입사일,급여를 출력하는 SCRIPT를 작성하시오.

|  |
| --- |
| Please enter the location name : chicago  old 6: AND loc = UPPER('&p\_name')  new 6: AND loc = UPPER('chicago')  DEPARTMENT EMPLOYEE START ANNUAL  NAME NAME DATE SALARY SALARY  -------------- ---------- ------------------ ------------ --------------  SALES BLAKE 01-MAY-81 $2,850.00 $34,200.00  MARTIN 28-SEP-81 $1,250.00 $15,000.00  ALLEN 20-FEB-81 $1,600.00 $19,200.00  TURNER 08-SEP-81 $1,500.00 $18,000.00  JAMES 03-DEC-81 $950.00 $11,400.00  WARD 22-FEB-81 $1,250.00 $15,000.00  6 rows selected. |