# SQL QUICK REFERENCE

## SELECT STATEMENT

```
SELECT * FROM EMP6

SELECT EMPNO, EMPNAME, EMPSALARY, GRADE, EMPID FROM EMP6

SELECT * FROM EMP6 WHERE EMPNO = 2

SELECT * FROM EMP6 WHERE EMPNO = 2 OR EMPNO = 4

SELECT EMPNAME, EMPSALARY FROM EMP6 WHERE EMPNO = 2 AND
EMPNAME LIKE 'd%'

SELECT EMPNAME FROM EMP6 WHERE EMPNAME = 'dinesh'

SELECT EMPNAME FROM EMP6 WHERE EMPNO != 3

SELECT EMPNAME FROM EMP6 WHERE EMPNO <> 3

SELECT * FROM EMP6 WHERE EMPSALARY > 10000

SELECT * FROM EMP6 WHERE EMPSALARY > 15000 AND EMPSALARY
<=20000

SELECT * FROM EMP6 WHERE EMPNAME >= 'di'

SELECT * FROM EMP6 WHERE EMPNAME != 'dinesh'

SELECT DISTINCT EMPSALARY FROM EMP6

SELECT ALL EMPNAME FROM EMP6

SELECT EMPNO ENO, EMPNAME NAME, EMPSALARY SAL FROM EMP6

SELECT ( EMPNO ) EMPLOYEENO FROM EMP6

SELECT EMPNO "EMPLOYEE NUMBER" FROM EMP6

SELECT EMPNO, EMPNAME, (EMPSALARY + 1000) SALARY FROM EMP6

SELECT EMPNO, EMPSALARY + 100 SAL FROMdineshkumar      Page 1
      2/20/2008 EMP6
SELECT (EMPSALARY + 1000) SAL FROM EMP6
```

```
SELECT EMPNO, -EMPSALARY FROM EMP6

SELECT (EMPSALARY + 100) - EMPSALARY FROM EMP6

SELECT EMPSALARY / 2 FROM EMP6

SELECT EMPNO, EMPNAME, EMPSALARY * 20 "NEW SAL" FROM EMP6

SELECT * FROM EMP6 WHERE EMPID IS NULL

SELECT * FROM EMP6 WHERE EMPNO IS NOT NULL

SELECT * FROM EMP6 WHERE EMPNAME LIKE '%ov%'

SELECT * FROM EMP6 WHERE EMPNAME LIKE 'sh%'

SELECT * FROM EMP6 WHERE EMPNAME LIKE '%an'

SELECT * FROM EMP6 WHERE EMPNAME LIKE '_i%'

SELECT * FROM EMP6 WHERE EMPNAME LIKE 'd_n%'

SELECT * FROM EMP6 WHERE EMPNAME LIKE 'dines_'

SELECT EMPNAME || GRADE FROM EMP6

SELECT EMPNAME || ',' || GRADE FROM EMP6

SELECT * FROM EMP6
UNION
SELECT * FROM EMP66

SELECT * FROM EMP6
UNION ALL
SELECT * FROM EMP66

SELECT * FROM EMP6
INTERSECT
SELECT * FROM EMP66

SELECT * FROM EMP6
MINUS
SELECT * FROM EMP66

SELECT * FROM EMP66
```

By Dinesh

**MINUS**
**SELECT * FROM EMP6**

**SELECT * FROM EMP6 WHERE EMPSALARY =10000 OR EMPSALARY = 15000 OR EMPSALARY =20000**

**SELECT * FROM EMP6 WHERE EMPSALARY IN (10000,15000,20000);**

**SELECT * FROM EMP6 WHERE EMPNAME LIKE 'dinesh' OR EMPNAME LIKE 'shovan'**

**SELECT * FROM EMP6 WHERE EMPNAME IN ('dinesh','shovan')**

**SELECT * FROM EMP6 WHERE EMPSALARY > 10000 AND EMPSALARY <= 20000**

**SELECT * FROM EMP6 WHERE EMPSALARY BETWEEN 10000 AND 20000**

**SELECT COUNT(*) FROM EMP6**

**SELECT COUNT(*) "TOTAL ROWS" FROM EMP6 WHERE EMPSALARY > 20000**

**SELECT COUNT(*) FROM EMP6 WHERE EMPSALARY/12 > 1000**

**SELECT COUNT(EMPSALARY) FROM EMP6**

**SELECT SUM(EMPSALARY) FROM EMP6**

**SELECT SUM(EMPSALARY) FROM EMP6 WHERE EMPSALARY > 15000**

**SELECT SUM(EMPSALARY) / SUM(EMPNO) FROM EMP6**

**SELECT AVG(EMPSALARY) FROM EMP6**

**SELECT SUM(EMPSALARY) / COUNT(*) "MANUAL", AVG(EMPSALARY) "AUTOMATIC" FROM EMP6**

**SELECT MAX(EMPSALARY) FROM EMP6**

**SELECT MAX(EMPNAME) FROM EMP6**

**SELECT MIN(EMPSALARY) FROM EMP6**

**SELECT MIN(EMPNAME) FROM EMP6**

By Dinesh

**SELECT VARIANCE(EMPSALARY) "SQUARE OF SD" FROM EMP6**

**SELECT STDDEV(EMPSALARY) "STD. DEVIATION" FROM EMP6**

**SELECT * FROM D**

**SELECT ST,ED, ADD_MONTHS(ED,3) "EXT DATE" FROM D**

**SELECT ED, ADD_MONTHS(ED,2) "NEW DATE" FROM D WHERE ST BETWEEN '4/1/1995' AND '12/21/1998'**

**SELECT ST,ED, LAST_DAY(ED) "MONTH LAST" FROM D**

**SELECT DISTINCT LAST_DAY('1-FEB-1995') " NON LEAP YR", LAST_DAY('1-FEB-1996') "LEAP YR" FROM D**

**SELECT MONTHS_BETWEEN(ST,ED) FROM D**

**SELECT MONTHS_BETWEEN(ED,ST) FROM D**

**SELECT DISTINCT SYSDATE FROM D**

**SELECT ABS(EMPSALARY) FROM EMP6**

**SELECT CEIL(EMPSALARY) FROM EMP6**

**SELECT COS(EMPSALARY),TAN(EMPSALARY),SIN(EMPSALARY) FROM EMP6**

**SELECT COS(100 * EMPSALARY) FROM EMP6**

**SELECT EXP(10) FROM EMP6**

**SELECT LN(EMPSALARY) FROM EMP6**

**SELECT LOG(EMPSALARY,10) FROM EMP6**

**SELECT MOD(EMPSALARY,EMPSALARY) FROM EMP6**

**SELECT POWER(10,2) FROM EMP6**

**SELECT SIGN(EMPSALARY) FROM EMP6**

**SELECT SIGN(-1) FROM EMP6**

**SELECT SIGN(0) FROM EMP6**

**SELECT SQRT(4) from emp6**

**SELECT CHR(EMPSALARY) FROM EMP6**

**SELECT CONCAT(EMPNAME,'XXX') FROM EMP6**

**SELECT INITCAP(EMPNAME) FROM EMP6**

**SELECT LOWER(EMPNAME), UPPER(EMPNAME) FROM EMP6**

**SELECT LPAD(EMPNAME,12,'#') "LPAD", RPAD( EMPNAME,10,'$') FROM EMP6**

**SELECT LPAD(EMPNAME,2) "LPAD", RPAD( EMPNAME,2) FROM EMP6**

**SELECT LPAD(RTRIM(EMPNAME),20,'*') FROM EMP6**

**SELECT LTRIM(EMPNAME,'s'), RTRIM(EMPNAME,'n') FROM EMP6**

**SELECT REPLACE(EMPNAME,'di','*@#') FROM EMP6**

**SELECT SUBSTR(EMPNAME,2,2) FROM EMP6**

**SELECT EMPNAME, INSTR(EMPNAME, 'o', 2) FROM EMP6**

**SELECT EMPNAME, LENGTH(EMPNAME) "LEN" from EMP6**

**SELECT TO_CHAR(EMPNO) FROM EMP6**

**SELECT LENGTH(TO_CHAR(EMPNO)) FROM EMP6**

**SELECT TO_NUMBER(EMPNO) FROM EMP6**

**SELECT GREATEST('DINESH','VINUSH','SHOVAN') "GREATEST", MAX(EMPNAME) "MAX", LEAST('DINESH','VINUSH') from EMP6**

**SELECT USER FROM EMP6**

## MATHEMATICAL FUNCTIONS

**ABS : the abs function returns the absolute value of a number.**

```
SELECT ABS(-23) FROM DUAL
```

**ACOS, ASIN, ATAN, ATAN2 : RETURN FOLLOWING CONVERSION VALUE**

```
SELECT ACOS(0.2), ASIN(.10), ATAN(.10) FROM DUAL
```

**AVG : the Avg function returns the average value of an expression.**

```
SELECT AVG(EMPSALARY) FROM EMP6
```

**BIN_TO_NUM : the bin_to_num function converts a bit vector to a number.**

```
SELECT BIN_TO_NUM(1,1,1) FROM DUAL
```

**BITAND : the bitand function returns an integer representing an AND operation on the bits of expr1 and expr2.**

```
SELECT BITAND(10,11) FROM DUAL
```

**CEIL : the ceil function returns the smallest integer value that is greater than or equal to a number.**

```
SELECT CEIL(1.90) FROM DUAL
```

**FLOOR : the floor function returns the largest integer value that is equal to or less than a number.**

```
SELECT CEIL(1.90) FROM DUAL
```

**CORR : the corr function returns the coefficient of correlation of a set of number pairs.**

```
SELECT CORR(1000,1) FROM DUAL
```

**COVAR_POP : the covar_pop function returns the population covariance of a set of number pairs.**

```
SELECT COVAR_POP(10,2) FROM DUAL
```

**COVAR_SAMP : the covar_samp function returns the sample covariance of a set of number pairs.**

**SELECT COVAR_SAMP(10,2) FROM DUAL**

**COUNT : The COUNT function returns the number of rows in a query.**

**SELECT COUNT(*) FROM EMP6**

**SELECT COUNT(DISTINCT EMPNAME) FROM EMP6**

**CUME_DIST : the cume_dist function returns the cumulative distribution of a value in a group of values. The cume_dist function will return a value that is >0 and <=1.**

**select cume_dist(1000) WITHIN GROUP (ORDER BY EMPsalary) from emp6;**

**select empNAME, EMPsalary, cume_dist() OVER (PARTITION BY EMPNO ORDER BY EMPsalary) from emp6**

**DENSE_RANK : the dense_rank function returns the rank of a row in a group of rows. It is very similar to the rank function. However, the rank function can cause non-consecutive rankings if the tested values are the same. Whereas, the dense_rank function will always result in consecutive rankings.**

**Used as an Aggregate Function**

**select dense_rank(1000) WITHIN GROUP (ORDER BY EMPsalary) from emP6**

**Used as an Analytic Function**

**select empNAME, EMPsalary, dense_rank() OVER (PARTITION BY EMPNO ORDER BY EMPsalary) from emp6**

**EXP : the exp function returns e raised to the nth power, where e = 2.71828183**

**SELECT EXP(10) FROM DUAL**

**EXTRACT : the extract function extracts a value from a date or interval value**

**SELECT EXTRACT(YEAR FROM DATE '2003-08-22') FROM DUAL**

**SELECT EXTRACT(MONTH FROM DATE '2003-08-22') FROM DUAL**

**GREATEST : the greatest function returns the greatest value in a list of expressions.**

**SELECT GREATEST (1,2,3,4) FROM DUAL**

**LEAST : the least function returns the smallest value in a list of expressions.**

**SELECT LEAST (1,2,3,4) FROM DUAL**

**LN : the ln function returns the natural logarithm of a number.**

**SELECT LN(10) FROM DUAL**

**LOG : the log function returns the logarithm of n base m.**

**SELECT LOG(10, 25) FROM DUAL**

**SELECT LOG(2, 25) FROM DUAL**

**MAX : The MAX function returns the maximum value of an expression.**

**SELECT MAX(EMPSALARY) FROM EMP6**

**MIN : The MIN function returns the minimum value of an expression.**

**SELECT MIN(EMPSALARY) FROM EMP6**

**MEDIAN : the median function returns the median of an expression.**

**SELECT MEDIAN(EMPSALARY) FROM EMP6**

**MOD : the mod function returns the remainder of m divided by n.**

**SELECT MOD(10,3) FROM DUAL**

**POWER : the power function returns m raised to the nth power.**

**SELECT POWER(10,2) FROM DUAL**

**REMAINDER : the remainder function returns the remainder of m divided by n.**

**SELECT REMAINDER(20,3) FROM DUAL**

**ROUND (NUM) : the round function returns a number rounded to a certain number of decimal places.**

**SELECT ROUND(124.99754) FROM DUAL**

**ROUND (DATE) : the round function returns a date rounded to a specific unit of measure.**

```
SELECT ROUND( TO_DATE('20-JAN-1986'), 'YEAR') FROM DUAL

SELECT ROUND( TO_DATE('20-JAN-1986'), 'MONTH') FROM DUAL
```

**SIGN : the sign function returns a value indicating the sign of a number.**

```
SELECT SIGN(10) from DUAL

SELECT SIGN(-10) FROM DUAL

SELECT SIGN(0) FROM DUAL
```

**SQRT : the sqrt function returns the square root of n.**

```
SELECT SQRT(4) FROM DUAL
```

**STDDEV : the stddev function returns the standard deviation of a set of numbers.**

```
SELECT STDDEV(EMPSALARY) FROM EMP6
```

**SUM : The SUM function returns the summed value of an expression.**

```
SELECT SUM(EMPSALARY) from EMP6
```

**TRUNC(NUM) : the trunc function returns a number truncated to a certain number of decimal places.**

```
SELECT TRUNC(1875.1234,1)FROM DUAL

SELECT TRUNC(1875.1234,2)FROM DUAL

SELECT TRUNC(1875.1234,3)FROM DUAL
```

**TRUNC(DATES) : the trunc function returns a date truncated to a specific unit of measure.**

```
SELECT TRUNC(TO_DATE('22-AUG-03'), 'YEAR') FROM DUAL

SELECT TRUNC(TO_DATE('22-AUG-03'), 'MONTH') FROM DUAL
```

**VAR_POP : the var_pop function returns the population variance of a set of numbers.**

```
SELECT VAR_POP(EMPSALARY) FROM EMP6
```

**VAR_SAMP : the var_samp function returns the sample variance of a set of numbers.**

```
SELECT VAR_SAMP(EMPSALARY) FROM EMP6
```

**VARIANCE : the variance function returns the variance of a set of numbers.**

```
SELECT VARIANCE(EMPSALARY) FROM EMP6
```

## ERROR FUNCTIONS

**SQLCODE : The SQLCODE function returns the error number associated with the most recently raised error exception. This function should only be used within the Exception Handling section of your code:**

**SQLERRM : The SQLERRM function returns the error message associated with the most recently raised error exception. This function should only be used within the Exception Handling section of your code:**

**You could use the SQLCODE function to raise an error as follows:**

```
---------------------------------------------------------------------------------------------
EXCEPTION
  WHEN OTHERS THEN
    raise_application_error(-20001,'An error was encountered - '||SQLCODE||' -
ERROR- '||SQLERRM);
END;
```

**Or you could log the error to a table as follows:**

```
EXCEPTION
  WHEN OTHERS THEN
    err_code := SQLCODE;
    err_msg := substr(SQLERRM, 1, 200);

    INSERT INTO audit_table (error_number, error_message) VALUES (err_code,
err_msg);
END;
```

---

## CONVERSION FUNCTIONS

**BIN_TO_NUM : the bin_to_num function converts a bit vector to a number.**

```
SELECT BIN_TO_NUM(1,1,1) FROM DUAL

SELECT BIN_TO_NUM(1,1,1,1) FROM DUAL
```

**CAST : the cast function converts one datatype to another.**

```
SELECT CAST('22-JAN-1986' AS CHAR(30)) FROM DUAL

SELECT CAST('111' AS INT) FROM DUAL
```

**CONVERSION POSSIBLITIES**
**CHAR/ VARCHAR2 ---->NUMBER,DATE ,RAW, ROWID**
**NUMBER -----> CHAR/VARCHAR2, NUMBER**
**DATE ------>CHAR/VARCHAR2**
**RAW -----> CHAR/VARCHAR2**
**ROWID ----> CHAR/VARCHAR2**
**NCHAR,NVARCHAR2 ----> NUMBER, DATE , RAW, ROWID**

**CHARTOROWID : the chartorowid function converts a char, varchar2, nchar, or nvarchar2 to a rowid.**

**The format of the rowid is:**

**BBBBBBB.RRRR.FFFFF**

**where:**

**BBBBBBB is the block in the database file;**
**RRRR is the row in the block;**
**FFFFF is the database file.**

**FROM_TZ : the from_tz function converts a TIMESTAMP value (given a TIME ZONE) to a TIMESTAMP WITH TIME ZONE value.**

```
select from_tz(TIMESTAMP '2005-09-11 01:50:42', '5:00') from dual;
```

**HEXTORAW : the hextoraw function converts a hexadecimal value into a raw value.**

```
SELECT HEXTORAW('443ED') FROM DUAL
```

**NUMTODSINTERVAL : the numtodsinterval function converts a number to an INTERVAL DAY TO SECOND literal.**

```
SELECT NUMTODSINTERVAL(1200,'DAY') FROM DUAL

SELECT NUMTODSINTERVAL(1500,'HOUR') FROM DUAL

SELECT NUMTODSINTERVAL(1500,'SECOND') FROM DUAL
```

**NUMTOYMINTERVAL : the numtoyminterval function converts a number to an INTERVAL YEAR TO MONTH literal.**

```
SELECT NUMTOYMINTERVAL(14000,'YEAR') FROM DUAL

SELECT NUMTOYMINTERVAL(14000,'MONTH') FROM DUAL
```

**RAWTOHEX : the rawtohex function converts a raw value into a hexadecimal value.**

```
SELECT RAWTOHEX('AB') FROM DUAL
```

**TO_CHAR : the to_char function converts a number or date to a string**

```
SELECT TO_CHAR('111') FROM DUAL
```

**TO_CLOB : the to_clob function converts a LOB value from the national character set to the database character set**

```
SELECT TO_CLOB(EMPNAME) from EMP6
```

**TO_DATE : the to_date function converts a string to a date.**

```
SELECT TO_DATE('09-07-1986','DD-MM-YYYY') FROM DUAL

SELECT TO_DATE('09071986','DDMMYY') FROM DUAL
```

**TO_DSINTERVAL : the to_dsinterval function converts a string to an INTERVAL DAY TO SECOND type.**

```
SELECT TO_DSINTERVAL('150 12:30:45') FROM DUAL
```

**TO_LOB : the to_lob function converts LONG or LONG RAW values to LOB values**

**TO_MULTI_BYTE : the to_multi_byte function returns a character value with all of the single-byte characters converted to multibyte characters. To use this function, your database character set contains both single-byte and multibyte characters.**

**TO_NCLOB : the to_nclob function converts a LOB value to a NCLOB value**

**TO_NCLOBE(COL.)**

**TO_NUMBER : the to_number function converts a string to a number.**

**SELECT TO_NUMBER('12345') FROM DUAL**

**TO_SINGLE_BYTE : the to_single_byte function returns a character value with all of the multibyte characters converted to single-byte characters. To use this function, your database character set contains both single-byte and multibyte characters.**

**select to_single_byte('Tech on the net') from dual;**

**TO_TIMESTAMP : the to_timestamp function converts a string to a timestamp**

**SELECT to_timestamp('2003/12/13 10:13:18', 'YYYY/MM/DD HH:MI:SS') FROM DUAL**

**TO_TIMESTAMP_TZ : the to_timestamp_tz function converts a string to a timestamp with time zone.**

**SELECT to_timestamp_tz('2003/12/13 10:13:18 -8:00', 'YYYY/MM/DD HH:MI:SS TZH:TZM') FROM DUAL**

**TO_YMINTERVAL : the to_yminterval function converts a string to an INTERVAL YEAR TO MONTH type.**

**SELECT to_yminterval('03-11') FROM DUAL**

# CHARACTER / STRING FUNCTIONS

**ASCII : the ascii function returns the NUMBER code that represents the specified character.**

```
SELECT ASCII('a') FROM DUAL
```

**ASCIISTR : the asciistr function converts a string in any character set to an ASCII string using the database character set.**

```
SELECT ASCIISTR('A B C') FROM DUAL
```

**CHR : the chr function is the opposite of the ascii function. It returns the character based on the NUMBER code.**

```
SELECT CHR(255) FROM DUAL
```

**COMPOSE : the compose function returns a Unicode string.**

```
SELECT COMPOSE('ABC') FROM DUAL
```

**CONCAT : the concat function allows you to concatenate two strings together.**

```
SELECT CONCAT('ABC','XYZ') FROM DUAL
```

**CONCAT (||) : the || operator allows you to concatenate 2 or more strings together**

```
SELECT 'ABC' || 'XYZ' FROM DUAL

SELECT 'ABC' || '--->' || 'XYZ' FROM DUAL
```

**DECOMPOSE : the decompose function accepts a string and returns a Unicode string.**

```
SELECT DECOMPOSE('A B C') FROM DUAL
```

**DUMP : the dump function returns a varchar2 value that includes the datatype code, the length in bytes, and the internal representation of the expression.**

```
SELECT DUMP('ABC',8) FROM DUAL
```

**SOUNDEX : the soundex function returns a phonetic representation (the way it sounds) of a string**

By Dinesh

**SELECT SOUNDEX('MUMMY') FROM DUAL**

**VSIZE : the vsize function returns the number of bytes in the internal representation of an expression.**

**SELECT VSIZE('HELLO HOW ARE YOU') FROM DUAL**

**INITCAP : the initcap function sets the first character in each word to uppercase and the rest to lowercase.**

**SELECT INITCAP('doctor') FROM DUAL**

**LENGTH : the length function returns the length of the specified string**

**SELECT LENGTH('HELLO HOW ARE YOU') FROM DUAL**

**LOWER : the lower function converts all letters in the specified string to lowercase. If there are characters in the string that are not letters, they are unaffected by this function.**

**SELECT LOWER('ABC') FROM DUAL**

**UPPER : the upper function converts all letters in the specified string to uppercase. If there are characters in the string that are not letters, they are unaffected by this function.**

**SELECT LOWER('abc') FROM DUAL**

**LPAD : the lpad function pads the left-side of a string with a specific set of characters.**

**SELECT LPAD('DINESH',20,'*') FROM DUAL**

**RPAD : the rpad function pads the right-side of a string with a specific set of characters**

**SELECT RPAD('DINESH',20,'*') FROM DUAL**

**LTRIM : the ltrim function removes all specified characters from the left-hand side of a string.**

**SELECT LTRIM('DINESH','D') FROM DUAL**

**RTRIM : the rtrim function removes all specified characters from the right-hand side of a string.**

By Dinesh

**SELECT RTRIM('DINESH','H') FROM DUAL**

**SUBSTR : the substr functions allows you to extract a substring from a string.**

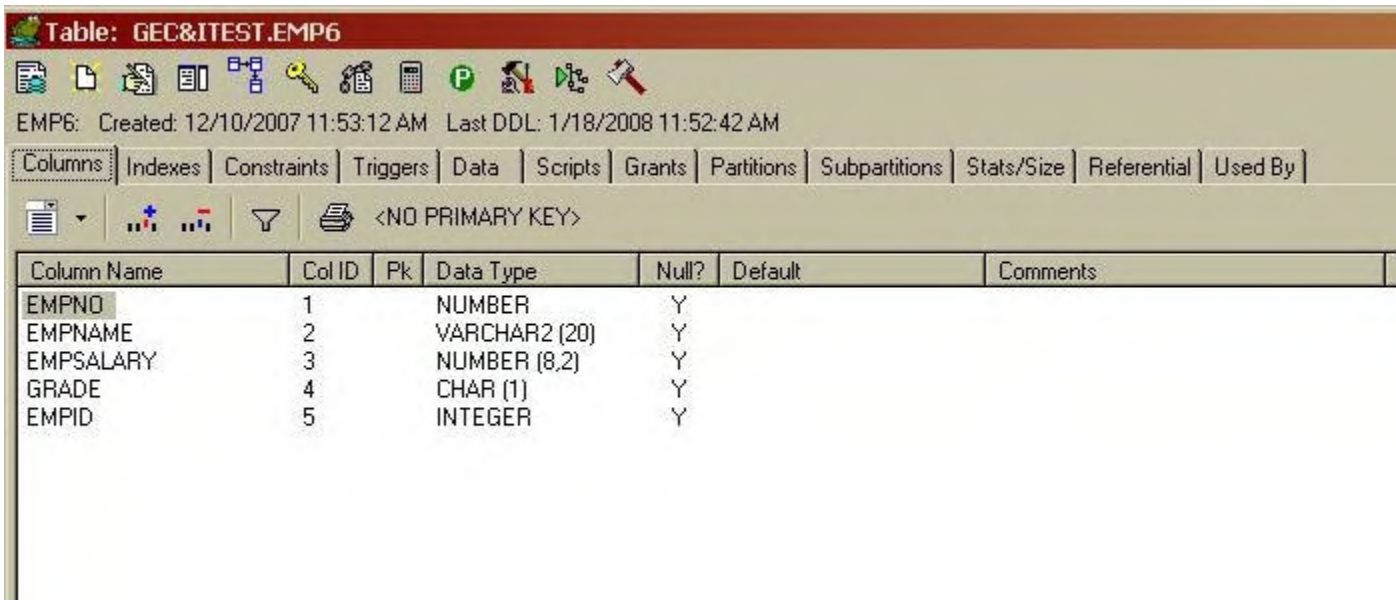
**SELECT SUBSTR('DINESH',3,3) FROM DUAL**

**TRANSLATE : the translate function replaces a sequence of characters in a string with another set of characters**

**SELECT TRANSLATE('12DINESH34','1234','@#$&') FROM DUAL**

**TABLES USED:**

Structure of emp6:




Structure of emp66:
Same as emp6

Data's in emp6 :

Data | Explain Plan | Auto Trace | DBMS Output | Code Statistics | Script Output

| EMPNO | EMPNAME | EMPSALARY | GRADE | EMPID |
|---|---|---|---|---|
| 1 | shovan | 22000 | d | |
| 2 | dinesh | 30000 | d | |
| 3 | senthil | 30000 | d | |
| 4 | srikanth | 15000 | d | |
| 5 | vishnu | 20000 | d | |

Data's in emp66

Data | Explain Plan | Auto Trace | DBMS Output | Code Statistics | Script Ou

| EMPNO | EMPNAME | EMPSALARY | GRADE | EMPID |
|---|---|---|---|---|
| 7 | ravi | 40000 | b | |
| 8 | balu | 40000 | c | |
| 9 | meena | 40000 | d | |
| 10 | rekha | 40000 | a | |
| 11 | suresh | 40000 | c | |
| 3 | senthil | 30000 | d | |

By Dinesh