ASSIGNMENT 7(SUBJECT: DBMS)

(Please use this format for doing assignments)

NAME OF STUDENT: Veeransh Shah

BRANCH: Computer Engineering

ID: 221070063

AIM: Study of Functions in SQL.

TOOL: MariaDB (If any other tool, please mention the name)

PROGRAMMING LANGUAGE: Structured Query language (SQL)

THEORY:

Explain Functions and Basic types of functions.

OUTPUT: (Copy Paste your output code here from your Mariadb Command Prompt, remove error lines form code)

LIST of FUNCTIONS:

ascii()	Returns the numeric value of the leftmost character of the string str. Returns 0 if str is the empty string. Returns NULL if str is NULL. ASCII() works for characters with numeric values from 0 to 255.
bin()	Returns a string representation of the binary value of N
char_length()	Returns the length of the string str measured in characters. A multi-byte character counts as a single character.
Concat()	Returns the string that results from concatenating the arguments. May have one or more arguments.
CONCAT_WS()	CONCAT_WS() stands for Concatenate With Separator and is a special form of CONCAT(). The first argument is the separator for the rest of the arguments. The separator is added between the strings to be concatenated. The separator can be a string, as can the rest of the arguments. If the separator is NULL, the result is NULL.

ELT(N,str1,str2,str3,...)	Returns str1 if N = 1, str2 if N = 2, and so on. Returns NULL if N is less than 1 or greater than the number of arguments. ELT() is the complement of FIELD().
UPPER(str)/ucase()	Returns the string str with all characters changed to uppercase according to the current character set mapping.
Lower()/lcase	Returns the string str with all characters changed to lowercase according to the current character set mapping.

Trim(){BOTH | LEADING | TRAILING **RTRIM(str)** **LTRIM(str)**	Returns the string str with all remstr prefixes or suffixes removed. If none of the specifiers BOTH, LEADING, or TRAILING is given, BOTH is assumed. remstr is optional and, if not specified, spaces are removed. Returns the string str with trailing space characters removed. Returns the string str with leading space characters removed.
SUBSTRING(str,pos) SUBSTRING(str FROM pos) SUBSTRING(str,pos,len) SUBSTRING(str FROM pos FOR len)	The forms without a len argument return a substring from string str starting at position pos. The forms with a len argument return a substring len characters long from string str, starting at position pos. The forms that use FROM are standard SQL syntax. It is also possible to use a negative value for pos. In this case, the beginning of the substring is pos characters from the end of the string, rather than the beginning. A negative value may be used for pos in any of the forms of this function.
STRCMP(str1, str2)	Compares two strings and returns 0 if both strings are equal, it returns -1 if the first argument is smaller than the second according to the current sort order otherwise it returns 1.
SPACE(N) RPAD(str,len,padstr) LPAD(str,len,padstr) RIGHT(str,len) LEFT(str,len)	Returns a string consisting of N space characters. Returns the string str, right-padded with the string padstr to a length of len characters. If str is longer than len, the return value is shortened to len characters. Returns the string str, left-padded with the string padstr to a length of len characters. If str is longer than len, the return value is shortened to len characters. Returns the rightmost len characters from the string str, or NULL if any argument is NULL. Returns the leftmost len characters from the string str, or NULL if any argument is NULL.
REVERSE(str)	Returns the string str with the order of the characters reversed.

REPLACE(str,from_str,to_str)	Returns the string str with all occurrences of the string from_str replaced by the string to_str. REPLACE() performs a case-sensitive match when searching for from_str.
REPEAT(str,count)	Returns a string consisting of the string str repeated count times. If count is less than 1, returns an empty string. Returns NULL if str or count are NULL.
LOCATE(substr,str)	Returns the position of the first occurrence of substring substr in string str.
LENGTH(str)	Returns the length of the string str, measured in bytes.
INSERT(str,pos,len,newstr)	Returns the string str, with the substring beginning at position pos and len characters long replaced by the string newstr. Returns the original string if pos is not within the length of the string. Replaces the rest of the string from position pos if len is not within the length of the rest of the string. Returns NULL if any argument is NULL.

INSTR(str,substr)	Returns the position of the first occurrence of substring substr in string str.
UNHEX(str)	Performs the inverse operation of HEX(str). That is, it interprets each pair of hexadecimal digits in the argument as a number and converts it to the character represented by the number. The resulting characters are returned as a binary string.

NOTE: please remember that order is important when dealing with combining/wrapping certain string functions.
For example:
This works:

```
1. SELECT UPPER(CONCAT(author_fname, ' ', author_lname)) AS "full name in caps"
2. FROM books;
```

While this does not:

```
1. SELECT CONCAT(UPPER(author_fname, ' ', author_lname)) AS "full name in caps"
2. FROM books;
```

UPPER only takes one argument and CONCAT takes two or more arguments, so they can't be switched in that way.
You could do it this way, however:

```
1. SELECT CONCAT(UPPER(author_fname), ' ', UPPER(author_lname)) AS "full name in caps"
2. FROM books;
```

But, the first example above would be better (more DRY*) because you wouldn't need to call UPPER two times.

CODE: String Function Challenges Solution

```
SELECT REVERSE(UPPER('Why does my cat look at me with such hatred?'));
SELECT UPPER(REVERSE('Why does my cat look at me with such hatred?'));
```

I-like-cats

```
SELECT REPLACE(CONCAT('I', ' ', 'like', ' ', 'cats'), ' ', '-');
SELECT REPLACE(title, ' ', '->') AS title FROM books;
```

```
1. SELECT
2. author_lname AS forwards,
3. REVERSE(author_lname) AS backwards
4. FROM books;
```

```
1. SELECT
2. UPPER
3. (
```



```
4. CONCAT(author_fname, ' ', author_lname)
5. ) AS 'full name in caps'
6. FROM books;
```

```
1. SELECT
2. CONCAT(title, ' was released in ', released_year) AS blurb
3. FROM books;
1. SELECT
2. title,
3. CHAR_LENGTH(title) AS 'character count'
4. FROM books;
```

```
1. SELECT
2. CONCAT(SUBSTRING(title, 1, 10), '...') AS 'short title',
3. CONCAT(author_lname, ',', author_fname) AS author,
4. CONCAT(stock_quantity, ' in stock') AS quantity
5. FROM books;
```

*********End of the Assignment*********