# ASSIGNMENT 3

**11) There is a popular story that someone whose last name is NULL has problems with automated systems such as driver's license systems, where her last name might not be accepted by the system. Would this happen with real applications? Explain your answer.**

Earlier systems lacked support for handling missing values. Hence when there are NULL values present, there might be difficulties in performing operations. To solve this problem, programmers introduced sentinel values that could be inserted in place of missing values. The sentinel values are recognized by computers and excludes it while computation.

In SQL, the developers allowed the possibility of null values. For Boolean operations, the system follows three-valued logic and for arithmetic operations, the system changes the null to mathematical zero.

Hence, In real world applications, the presence of null value cannot cause error.

**12) Write a SQL statement to count the number of rows in the example table of employees and count the number of values of commission. Show and explain your results.**

```
SELECT COUNT(*) AS TOTALROWS, COUNT(COMM) AS NONNULLCOMMISSION
FROM EMPLOYEES;
```

| TOTALROWS | NONNULLCOMMISSION |
|---|---|
| 6 | 2 |

Here,
COUNT(*) will return the total number of rows from table employees table.
COUNT(COMM): returns the number of non-null values in the "COMM" column

**13) Write a SQL statement to compute the total of commissions using SUM, without the null value function, and with the null value function. Show and explain your results.**

```
-- Sum without NULL value function

SELECT SUM(COMM) AS TOTALCOMMISSIONWITHOUTNULL
FROM EMPLOYEES;
```

| TOTALCOMMISSIONWITHOUTNULL |
|---|
| 115 |

-- Sum with NULL value function

```
SELECT SUM(NVL(COMM, 0)) AS TOTALCOMMISSIONWITHNULL
FROM EMPLOYEES;
```

| TOTALCOMMISSIONWITHNULL |
|---|
| 115 |

The reason we are getting the same result is because the aggregate functions, namely (SUM, AVG, COUNT etc.) omit the NULL values during computation. Hence if we Omit the values or change them to zero, the answer would be the same.

**14) Are the set operations of SQL necessary? That is, are there any queries that can be written using set functions that cannot be expressed without set operations?**

Set operations are not entirely necessary. The functionality of these operations can be achieved by join commands. The Intersect can be replaced by inner join, Union can be replaced by full outer join and Except can be replaced by left/right outer join.

**15) Use set operations to find all of the employees who do not earn commission.**

```
SELECT EMP# FROM EMPLOYEES
MINUS
SELECT EMP# FROM EMPLOYEES WHERE COMM IS NOT NULL;
```

| EMP# |
|---|
| 20 |
| 30 |
| 40 |
| 70 |

**16) Use a single SELECT statement without set operations to derive the same result.**

```
SELECT EMP# FROM EMPLOYEES
WHERE COMM IS NULL;
```

| EMP# |
|---|
| 20 |
| 30 |
| 40 |
| 70 |

**17) Use set operations to find all the departments located in New York City or Los Angeles. Also use a single SELECT statement to derive the same result.**

-- Using set operations

```
SELECT DNAME, LOC FROM DEPARTMENTS
INTERSECT
SELECT DNAME, LOC FROM DEPARTMENTS WHERE LOC='NYC' OR LOC='LA';
```

| DNAME | LOC |
|---|---|
| FINANCE | NYC |
| MARKETING | NYC |
| SALES | LA |

-- Single SELECT Statement

```
SELECT DNAME, LOC FROM DEPARTMENTS WHERE LOC='NYC' OR LOC='LA';
```

| DNAME | LOC |
|---|---|
| FINANCE | NYC |
| SALES | LA |
| MARKETING | NYC |

**18) List a table of employee names and their salary grade number.**

```
SELECT E.ENAME, S.GRADE#
FROM EMPLOYEES E
JOIN SALGRADE S ON E.SAL BETWEEN S.LOWSAL AND S.HISAL;
```

| ENAME | GRADE# |
|---|---|
| SMITH | 6 |
| SMITH | 5 |
| CHEN | 4 |
| LIU | 4 |
| LIU | 3 |
| FOONG | 3 |
| JONES | 1 |
| JAMES | 1 |

**19) Find all the orders placed by customers who are not in the customer list.**

```
SELECT O.ORDER# FROM ORDERS O
LEFT JOIN CUSTOMERS C ON O.CUST# = C.CUST#
WHERE C.CUST# IS NULL;
```

| ORDER# |
|---|
| 105 |
| 38 |

**20) Find all customers who have not placed orders.**

```
SELECT C.CUST# FROM CUSTOMERS C
LEFT JOIN ORDERS O ON C.CUST# = O.CUST#
WHERE O.CUST# IS NULL;
```

| CUST# |
|---|
| 57 |
| 540 |