Data Engineering Day 06

The credit for this course goes to Coursera. Click More

Another link : Azure data Engineer

Design database including (ERD)

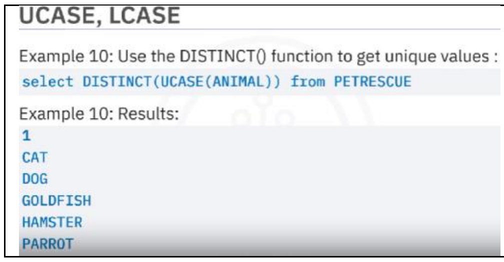
# creating two tables using SQL commands “create”.

```
CREATE TABLE PETSALE (
ID INTEGER NOT NULL,
PET CHAR (20),
SALEPRICE DECIMAL (6,2),
PROFIT DECIMAL (6,2),
SALEDATE DATE
);


CREATE TABLE PET (
ID INTEGER NOT NULL,
ANIMAL VARCHAR (20),
QUANTITY INTEGER
);
```

```
SELECT DEP_ID, COUNT (*)
FROM EMPLOYEES
GROUP BY DEP_ID.
```

# some more advanced commands used in Database.



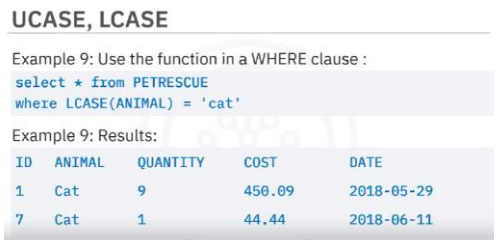
UCASE, LCASE

Example 10: Use the DISTINCT() function to get unique values :

```
select DISTINCT(UCASE(ANIMAL)) from PETRESCUE
```

Example 10: Results:

```
1
CAT
DOG
GOLDFISH
HAMSTER
PARROT
```



UCASE, LCASE

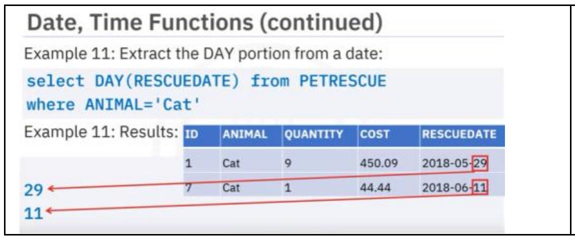
Example 9: Use the function in a WHERE clause :

```
select * from PETRESCUE
where LCASE(ANIMAL) = 'cat'
```

Example 9: Results:

```
ID   ANIMAL    QUANTITY    COST      DATE
1    Cat       9           450.09    2018-05-29
7    Cat       1           44.44     2018-06-11
```

# Date and time functions:


Date, Time Functions (continued)

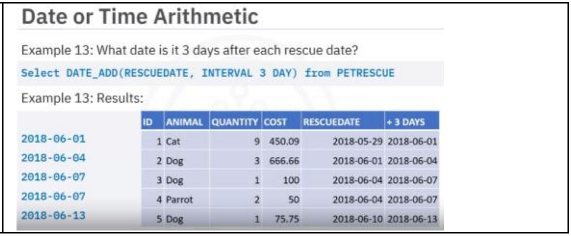
Example 11: Extract the DAY portion from a date:

```
select DAY(RESCUEDATE) from PETRESCUE
where ANIMAL='Cat'
```

Example 11: Results:

ID	ANIMAL	QUANTITY	COST	RESCUEDATE
1	Cat	9	450.09	2018-05-29
7	Cat	1	44.44	2018-06-11

```
29
11
```


Date or Time Arithmetic

Example 13: What date is it 3 days after each rescue date?

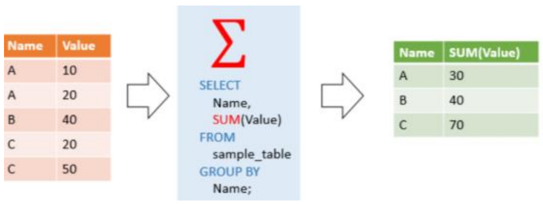
```
Select DATE_ADD(RESCUEDATE, INTERVAL 3 DAY) from PETRESCUE
```

Example 13: Results:

```
2018-06-01
2018-06-04
2018-06-07
2018-06-07
2018-06-13
```

ID	ANIMAL	QUANTITY	COST	RESCUEDATE	+ 3 DAYS
1	Cat	9	450.09	2018-05-29	2018-06-01
2	Dog	3	666.66	2018-06-01	2018-06-04
3	Dog	1	100	2018-06-04	2018-06-07
4	Parrot	2	50	2018-06-04	2018-06-07
5	Dog	1	75.75	2018-06-10	2018-06-13

# Sum, Avg functions in MySQL:

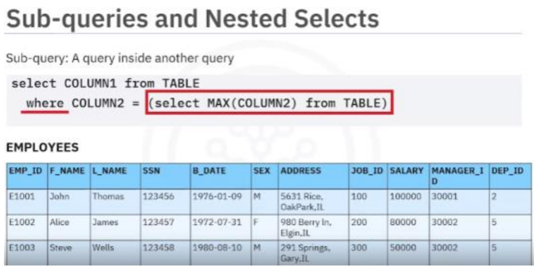


Name	Value
A	10
A	20
B	40
C	20
C	50

```
SELECT
    Name,
    SUM(Value)
FROM
    sample_table
GROUP BY
    Name;
```

Name	SUM(Value)
A	30
B	40
C	70

# Nested Queries from MySQL:


Sub-queries and Nested Selects

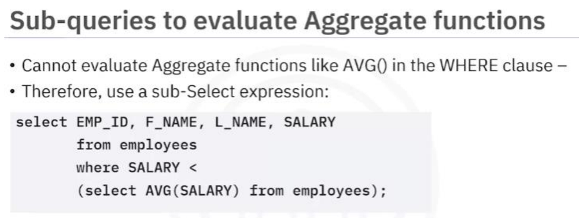
Sub-query: A query inside another query

```
select COLUMN1 from TABLE
  where COLUMN2 = (select MAX(COLUMN2) from TABLE)
```

EMPLOYEES

EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID
E1001	John	Thomas	123456	1976-01-09	M	5631 Rice, OakPark,IL	100	100000	30001	2
E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000	30002	5
E1003	Steve	Wells	123458	1980-08-10	M	291 Springs, Gary,IL	300	50000	30002	5

*In the figure shown below, from the given table employees, it will calculate the salary less than average salary and finally returns the output as employee_ID, F_Name, L_Name,Salary.


Sub-queries to evaluate Aggregate functions

- Cannot evaluate Aggregate functions like AVG() in the WHERE clause –
- Therefore, use a sub-Select expression:

```
select EMP_ID, F_NAME, L_NAME, SALARY
       from employees
       where SALARY <
       (select AVG(SALARY) from employees);
```

# Some of the practice questions:

```
CREATE TABLE EMPLOYEES (
                            EMP_ID CHAR (9) NOT NULL,
                            F_NAME VARCHAR (15) NOT NULL,
                            L_NAME VARCHAR (15) NOT NULL,
                            SSN CHAR (9),
                            B_DATE DATE,
                            SEX CHAR,
                            ADDRESS VARCHAR (30),
                            JOB_ID CHAR (9),
                            SALARY DECIMAL (10,2),
                            MANAGER_ID CHAR (9),
                            DEP_ID CHAR (9) NOT NULL,
                            PRIMARY KEY (EMP_ID));
```

```
  CREATE TABLE JOB_HISTORY (
                          EMPL_ID CHAR(9) NOT NULL,
                          START_DATE DATE,
                          JOBS_ID CHAR(9) NOT NULL,
                          DEPT_ID CHAR(9),
                          PRIMARY KEY (EMPL_ID,JOBS_ID));

 CREATE TABLE JOBS (
                          JOB_IDENT CHAR(9) NOT NULL,
                          JOB_TITLE VARCHAR(30),
                          MIN_SALARY DECIMAL(10,2),
                          MAX_SALARY DECIMAL(10,2),
                          PRIMARY KEY (JOB_IDENT));

CREATE TABLE DEPARTMENTS (
                          DEPT_ID_DEP CHAR(9) NOT NULL,
                          DEP_NAME VARCHAR(15) ,
                          MANAGER_ID CHAR(9),
                          LOC_ID CHAR(9),
                          PRIMARY KEY (DEPT_ID_DEP));

CREATE TABLE LOCATIONS (
                          LOCT_ID CHAR(9) NOT NULL,
                          DEP_ID_LOC CHAR(9) NOT NULL,
                          PRIMARY KEY (LOCT_ID,DEP_ID_LOC));
```