**Module 3 – INTERMEDIATE SQL**

**REFINING YOUR RESULTS**

**Exercise 1: String Patterns**

SELECT F\_NAME , L\_NAME

FROM EMPLOYEES

WHERE ADDRESS LIKE '%Elgin,IL%';

SELECT F\_NAME , L\_NAME

FROM EMPLOYEES

WHERE B\_DATE LIKE '197%';

SELECT \*

FROM EMPLOYEES

WHERE (SALARY BETWEEN 60000 AND 70000) AND DEP\_ID = 5;

**Exercise 2: Sorting**  
 **SELECT F\_NAME, L\_NAME, DEP\_ID**

FROM EMPLOYEES

ORDER BY DEP\_ID;

SELECT F\_NAME, L\_NAME, DEP\_ID

FROM EMPLOYEES

ORDER BY DEP\_ID DESC, L\_NAME DESC;

SELECT D.DEP\_NAME , E.F\_NAME, E.L\_NAME

FROM EMPLOYEES as E, DEPARTMENTS as D

WHERE E.DEP\_ID = D.DEPT\_ID\_DEP

ORDER BY D.DEP\_NAME, E.L\_NAME DESC;

**Exercise 3: Grouping**

SELECT DEP\_ID, COUNT(\*)

FROM EMPLOYEES

GROUP BY DEP\_ID;

SELECT DEP\_ID, COUNT(\*), AVG(SALARY)

FROM EMPLOYEES

GROUP BY DEP\_ID;

SELECT DEP\_ID, COUNT(\*) AS "NUM\_EMPLOYEES", AVG(SALARY) AS "AVG\_SALARY"

FROM EMPLOYEES

GROUP BY DEP\_ID;

SELECT DEP\_ID, COUNT(\*) AS "NUM\_EMPLOYEES", AVG(SALARY) AS "AVG\_SALARY"

FROM EMPLOYEES

GROUP BY DEP\_ID

ORDER BY AVG\_SALARY;

SELECT DEP\_ID, COUNT(\*) AS "NUM\_EMPLOYEES", AVG(SALARY) AS "AVG\_SALARY"

FROM EMPLOYEES

GROUP BY DEP\_ID

HAVING count(\*) < 4

ORDER BY AVG\_SALARY;

**Functions, Multiple Tables, and Sub-queries**

**Exercise 2: Aggregate Functions**

SELECT SUM(COST) FROM PETRESCUE

SELECT SUM(COST) AS SUM\_OF\_COST FROM PETRESCUE

SELECT MAX(QUANTITY) FROM PETRESCUE

SELECT AVG(COST) FROM PETRESCUE

SELECT AVG(COST) FROM PETRESCUE WHERE ANIMAL = 'DOG' (or)

select AVG(COST/QUANTITY) from PETRESCUE where ANIMAL = 'Dog';

**Exercise 3: Scalar and String Functions**

SELECT ROUND(COST) FROM PETRESCUE

SELECT LENGTH(ANIMAL) FROM PETRESCUE

SELECT UCASE(ANIMAL) FROM PETRESCUE

SELECT DISTINCT(UCASE(ANIMAL)) FROM PETRESCUE

SELECT \* FROM `PETRESCUE` WHERE LCASE(ANIMAL) = 'CAT'

**Exercise 4: Date and Time Functions**

SELECT MONTH(RESCUEDATE) FROM PETRESCUE

SELECT SUM(QUANTITY) FROM PETRESCUE WHERE MONTH(RESCUEDATE) = 5

SELECT SUM(QUANTITY) FROM PETRESCUE WHERE DAY(RESCUEDATE) = 14

SELECT DATE\_ADD(RESCUEDATE, INTERVAL 3 DAY) AS third\_day FROM `PETRESCUE`

SELECT DATEDIFF(CURDATE(), RESCUEDATE) AS length\_of\_time FROM `PETRESCUE`

**Module 6**

**ACID**:

--#SET TERMINATOR @

CREATE PROCEDURE TRANSACTION\_ROSE -- Name of this stored procedure routine

LANGUAGE SQL -- Language used in this routine

MODIFIES SQL DATA -- This routine will only write/modify data in the table

BEGIN

DECLARE SQLCODE INTEGER DEFAULT 0; -- Host variable SQLCODE declared and assigned 0

DECLARE retcode INTEGER DEFAULT 0; -- Local variable retcode with declared and assigned 0

DECLARE CONTINUE HANDLER FOR SQLEXCEPTION -- Handler tell the routine what to do when an error or warning occurs

SET retcode = SQLCODE; -- Value of SQLCODE assigned to local variable retcode

UPDATE BankAccounts

SET Balance = Balance-200

WHERE AccountName = 'Rose';

UPDATE BankAccounts

SET Balance = Balance+200

WHERE AccountName = 'Shoe Shop';

UPDATE ShoeShop

SET Stock = Stock-1

WHERE Product = 'Boots';

UPDATE BankAccounts

SET Balance = Balance-300

WHERE AccountName = 'Rose';

IF retcode < 0 THEN -- SQLCODE returns negative value for error, zero for success, positive value for warning

ROLLBACK WORK;

ELSE

COMMIT WORK;

END IF;

END

@ -- Routine termination character

CALL TRANSACTION\_ROSE; -- Caller query

SELECT \* FROM BankAccounts;

SELECT \* FROM ShoeShop;