MySQL - RDBMS

Agenda

- DQL - SELECT
    - WHERE clause
        - IN operator
        - BETWEEN operator
        - LIKE operator
- DML - UPDATE
- DML - DELETE
- DDL - TRUNCATE
- DDL - DROP
- HELP
- DUAL table
- SQL Functions
    - String Functions

WHERE clause

- terminal> mysql -u sunbeam -psunbeam classwork
- IN operator
    - Similar to Logical OR for equality checking with multiple values (for same column).
    - NOT IN operator -- inverse of IN operator

```
SELECT USER(), DATABASE();
-- | sunbeam@localhost | classwork  |

-- display all MANAGERS and emps from dept 10.
SELECT * FROM emp WHERE job = 'MANAGER' OR deptno = 10;

-- display all emps whose sal is less than 1000 or sal is more than 3500.
SELECT * FROM emp WHERE sal < 1000 OR sal > 3500;

-- display all analysts and managers.
SELECT * FROM emp WHERE job = 'ANALYST' OR job = 'MANAGER';

SELECT * FROM emp WHERE job IN ('ANALYST', 'MANAGER');

-- display emps whose names are JAMES, KING, MARTIN, FORD.
SELECT * FROM emp WHERE ename = 'JAMES' OR ename = 'KING' OR ename = 'MARTIN' OR
ename = 'FORD';

SELECT * FROM emp WHERE ename IN ('JAMES', 'KING', 'MARTIN', 'FORD');

-- display all emps whose are not salesman or manager.
SELECT * FROM emp WHERE job NOT IN ('SALESMAN', 'MANAGER');
SELECT * FROM emp WHERE NOT (job IN ('SALESMAN', 'MANAGER'));

SELECT * FROM emp WHERE NOT (job = 'SALESMAN' OR job = 'MANAGER');
```

- BETWEEN operator
  - range check (including both ends).
  - col BETWEEN start AND end
    - col >= start AND col <= end.

```
-- display manager of dept 20.
SELECT * FROM emp WHERE job = 'MANAGER' AND deptno = 20;

-- display all emps in dept 30 whose sal is less than 1500.
SELECT * FROM emp WHERE deptno = 30 AND sal < 1500;

-- display all emps in sal range 1500 to 3000.
SELECT * FROM emp WHERE sal >= 1500 AND sal <= 3000;

SELECT * FROM emp WHERE sal BETWEEN 1500 AND 3000;

-- display all emps hired in year 1982.
SELECT * FROM emp WHERE hire >= '1982-01-01' AND hire <= '1982-12-31';

SELECT * FROM emp WHERE hire BETWEEN '1982-01-01' AND '1982-12-31';
```

- Paperwork
  - ADAM
  - B
  - BLAKE
  - CLARK
  - J
  - JAMES
  - KING

```
INSERT INTO emp(ename) VALUES ('B'), ('J');

SELECT ename FROM emp ORDER BY ename;

-- display all emps whose names start with 'B' to 'J';
SELECT ename FROM emp WHERE ename BETWEEN 'B' AND 'J';
-- JAMES & JONES are not displayed because alphabetically they come after J.

SELECT ename FROM emp WHERE ename BETWEEN 'B' AND 'K';
-- will display all names starting from B to J + will also display "K" (if
present).

SELECT ename FROM emp WHERE ename BETWEEN 'B' AND 'K' AND ename != 'K';
```

```
-- will display all names starting from B to J

-- display all emps whose sal is not in range 1000 to 2000.
SELECT * FROM emp WHERE sal NOT BETWEEN 1000 AND 2000;

-- display all emps whose sal is not in range 1500 to 3000.
SELECT * FROM emp WHERE sal NOT BETWEEN 1500 AND 3000;

-- display all emps between B to K and T to Z;
SELECT * FROM emp WHERE
ename BETWEEN 'B' AND 'K'
OR
ename BETWEEN 'T' AND 'Z';
```

- LIKE operator.
  - Used with strings for finding similar records.
  - Wildcard character to give pattern.
    - % : Any number of any char or Empty.
    - _ : Single occurrence of any char.
  - NOT LIKE : inverse of LIKE

```
-- find all emps whose name start with M.
SELECT * FROM emp WHERE ename LIKE 'M%';

-- find all emps whose name ends with H.
SELECT * FROM emp WHERE ename LIKE '%H';

-- find all emps whose name contain U.
SELECT * FROM emp WHERE ename LIKE '%U%';

-- find all emps whose name contains A twice.
SELECT * FROM emp WHERE ename LIKE '%A%A%';
-- ADAMS --> YES --> %=, %=D, %=MS
-- WARD --> NO
-- ANNA --> YES --> %=, %=NN, %=
-- RAAM --> YES --> %=R, %=, %=M

-- find all emps whose name is start with S to Z.
SELECT * FROM emp WHERE ename BETWEEN 'S' AND 'Z'
OR ename LIKE 'Z%';
```

```
-- display emps having 4 letter name.
SELECT * FROM emp WHERE ename LIKE '____';

-- display all emps whose name contains R on 3rd position.
SELECT * FROM emp WHERE ename LIKE '__R%';
```

```
-- display emps with 4 letter name and 3rd pos is R
SELECT * FROM emp WHERE ename LIKE '__R_';
```

- SELECT cols FROM tablename WHERE condition ORDER BY cols LIMIT n;

```
-- display emp with highest sal in range 1000 to 2000.
SELECT * FROM emp WHERE sal BETWEEN 1000 AND 2000
ORDER BY sal DESC LIMIT 1;

-- display CLERK with min sal.
SELECT * FROM emp
WHERE job = 'CLERK'
ORDER BY sal
LIMIT 1;

-- diplay fifth lowest sal from dept 20 and 30.
SELECT sal FROM emp WHERE deptno IN (20, 30);

SELECT sal FROM emp WHERE deptno IN (20, 30)
ORDER BY sal;

SELECT DISTINCT sal FROM emp
WHERE deptno IN (20, 30)
ORDER BY sal LIMIT 4, 1;
```

DML - UPDATE

```
SHOW TABLES;

SELECT * FROM stud;

-- change marks to 95.00 for student with id 7.
UPDATE stud SET marks = 95.80 WHERE id = 7;

SELECT * FROM stud;

SELECT id,name,subject,price FROM books;

-- increase price of C Programming books by 50.
UPDATE books SET price = price + 50
WHERE subject = 'C Programming';

SELECT id,name,subject,price FROM books;

-- increase price of all books by 5%.
UPDATE books SET price = price + price * 0.05;

SELECT id,name,subject,price FROM books;
SELECT * FROM emp;

UPDATE emp SET empno=1, sal=1000, comm=NULL WHERE ename = 'B';

SELECT * FROM emp;
```

DML - DELETE

```
-- delete given rows from emp table -- ename = B / J
DELETE FROM emp WHERE ename IN ('B', 'J');

SELECT * FROM emp;

SELECT * FROM newemp;

-- delete all rows from emp table
DELETE FROM newemp;

SELECT * FROM newemp;

DESCRIBE newemp;

-- delete price of book with id 1001 -- edit/update
UPDATE books SET price = NULL WHERE id = 1001;

SELECT * FROM books;
```

DDL - TRUNCATE

- TRUNCATE is DDL query.
- TRUNCATE is to delete all rows - cannot use WHERE clause.
- Table structure is not deleted (simiar to DELETE query).

```
-- delete all books
TRUNCATE TABLE books;

SELECT * FROM books;

DESCRIBE books;
```

DDL - DROP

- DROP command can be used for dropping/deleting whole table or database.
  - DROP TABLE tablename;
  - DROP DATABASE dbname;

```
SELECT * FROM dummy;

DESCRIBE dummy;

DROP TABLE dummy;

SHOW TABLES;

DESCRIBE dummy;
-- error
```

DELETE vs TRUNCATE vs DROP

- DELETE
    - Used to delete rows (not structure).
    - All rows or as per WHERE condition.
    - DML operation
    - DML ops can be rollbacked (undo/discard) using transaction.
    - In most RDBMS, slower operation.
- TRUNCATE
    - Used to delete rows (not structure).
    - All rows.
    - DDL operation.
    - DDL ops cannot be rollbacked.
    - In most RDBMS, faster operation.
- DROP
    - Used to delete rows as well as struct.
    - DDL operation.
    - DDL ops cannot be rollbacked.
    - This is fastest operation.

HELP

```
HELP SELECT;

HELP INSERT;

HELP Functions;

HELP String Functions;

HELP UPPER;

SELECT UPPER('MySql');

SELECT SUBSTRING('SUNBEAM', 4, 2);
```

DUAL Table

- In Oracle, DUAL table.
  - DUAL table is in memory single row single column virtual table to support SELECT query syntax.
  - SELECT cols FROM tablename;

```
SELECT 2 + 4 * 5 FROM DUAL;

SELECT VERSION() FROM DUAL;

SELECT NOW() FROM DUAL;

SELECT LOWER('SunBeam') FROM DUAL;
```

- One cannot manipulate or drop DUAL table.

```
DESCRIBE DUAL;
-- error

DROP TABLE DUAL;
-- error

DELETE FROM DUAL;
-- error
```

- ANSI SQL influenced by Oracle SQL.
- DUAL table is added in ANSI standard.
- In ANSI SQL, this table is optional.

```
SELECT 2 + 4 * 5;

SELECT VERSION();

SELECT NOW();

SELECT LOWER('SunBeam');
```

SQL Functions

String Functions

```
SELECT LOWER('India');

SELECT ename, LOWER(ename) FROM emp;
SELECT LEFT('Sunbeam', 2), RIGHT('Sunbeam', 2);

SELECT ename, LEFT(ename, 2), RIGHT(ename, 2) FROM emp;

-- display all emps whose name start with B to K.
SELECT ename, LEFT(ename,1) FROM emp
WHERE LEFT(ename,1) BETWEEN 'B' AND 'K';

SELECT SUBSTRING('SUNBEAM', 2, 3);
-- UNB (+ve pos -- pos from left)

SELECT SUBSTRING('SUNBEAM', -5, 3);
-- NBE (-ve pos -- pos from right)

SELECT SUBSTRING('SUNBEAM', 4, 0);
-- len=0, means no chars after given pos - empty

SELECT SUBSTRING('SUNBEAM', 4, -2);
-- len < 0, means no chars after given pos - empty

-- print chars 2-5 for all emp names.
SELECT ename, SUBSTRING(ename, 2, 4) FROM emp;

SELECT CONCAT('SUNBEAM', ' ', 'INFOTECH');

SELECT CONCAT('SUNBEAM', 2021);

SELECT CONCAT(ename, ' - ', job) FROM emp;

SELECT CONCAT(ename, ' is working in dept ', deptno, ' as ', job, '.') FROM emp;

SELECT LENGTH('    abcd   ');

SELECT TRIM('    abcd   ');

SELECT LENGTH(TRIM('    abcd   '));
-- output of TRIM('    abcd   ') is given as input to LENGTH().

SELECT LPAD('Sunbeam', 10, '*');
SELECT RPAD('Sunbeam', 10, '*');

SELECT RPAD(LPAD('Sunbeam', 10, '*'), 13, '*');
```