DATABASE MANAGEMENT SYSTEM

Subash Manandhar

Chapter 3 – Structured Query Language (SQL)

- SQL (Structured Query Language) is the database language designed for managing data in RDBMS (Relational Database Management System).
- Its scope includes data insert, query, update and delete, schema creation and modification and data access control.
- It is ANSI (American National Standard Institute) standard.
- Can execute queries against database.
- Different types of SQL statements are:
  - DDL (Data Definition Language)
    - CREATE, ALTER, DROP, TRUNCATE
  - DML (Data Manipulation Language)
    - INSERT, UPDATE, DELETE
  - DQL (Data Query Language)
    - SELECT
  - TCL (Transaction Control Language)
    - COMMIT, ROLLBACK
  - DCL (Data Control Language)
    - GRANT, REVOKE

Chapter 3 – Structured Query Language (SQL)

- SEQUEL (Structured English Query Language) in 1970's, developed at IBM.
- SQL-86 (First ANSI standard)
- SQL-89, SQL 1
- SQL-92, SQL 2 (Most of dbms use it)
- SQL-99, SQL 3 (recursive queries, triggers and OO features)
- SQL-2003, windows functions, XML related features
- SQL-2006, XML query support (Xquery)
- SQL-2008
- SQL-2011, improved support for temporal database.

Chapter 3 – Structured Query Language (SQL)

- **DDL (Data Definition Language)**
  - used to define the database schema.
  - used to create and modify the structure of database objects in the database.
  - **To create database:**
    - Syntax:
      - CREATE DATABASE dbname;
  - **To create table:**
    - Syntax:
      - **CREATE TABLE tablename**
        (
        column1 datatype(size),
        column2 datatype(size),
        …….
        );

CREATE TABLE tablename
(
column1 datatype(size) Constraints,
column2 datatype(size),
…….
);

- e.g.
  - CREATE DATABASE cms
  - CREATE TABLE student

```
(
    S_ID int,
    name varchar (20),
    address varchar(20)
)
```

Chapter 3 – Structured Query Language (SQL)

- CREATE TABLE student

(

S_ID int PRIMARY KEY,

name varchar (20) NOT NULL,

address varchar(20)

);

- CREATE TABLE student

(

S_ID int ,

name varchar (20) NOT NULL,

address varchar(20),

CONSTRAINTS pk_sid PRIMARY KEY(S_ID)

)

- CREATE TABLE student

(

S_ID int ,

name varchar (20) NOT NULL,

address varchar(20),

CONSTRAINTS pk_sid PRIMARY KEY(S_ID, name)

)

Chapter 3 – Structured Query Language (SQL)

- **To alter table:**
- Syntax:
  - ALTER TABLE tablename
    ADD columnname datatype
  - ALTER TABLE tablename
    DROP COLUMN columnname
  - ALTER TABLE tablename
    ALTER COLUMN columnname datatype

- e.g.
- ALTER TABLE student
  ADD phone varchar(10)
- ALTER TABLE student
  DROP COLUMN address
- ALTER TABLE student
  ALTER COLUMN phone integer

ALTER TABLE student
ADD CONSTRAINT pk_sid PRIMARY KEY (S_ID)

ALTER TABLE student
DROP CONSTRAINT pk_sid

Chapter 3 – Structured Query Language (SQL)

- **To delete table:**
- Syntax:
  - DROP TABLE tablename
- **To delete data inside table, not table itself:**
- Syntax:
  - TRUNCATE TABLE tablename
- e.g.
- DROP TABLE student

- TRUNCATE TABLE student

- **To rename table:**
- Syntax:
  - ALTER TABLE tablename
    RENAME TO newtablename

  e.g.

  ALTER TABLE xyz

  RENAME TO abc

- **To rename column:**
- Syntax:
  - ALTER TABLE tablename
    CHANGE oldname newname datatype

  e.g.

  ALTER TABLE student

  CHANGE phone contact_no integer

Chapter 3 – Structured Query Language (SQL)

- **DML (Data Manipulation Language)**
  - Used to insert , update, delete data in database.
- **To insert data in table:**
- Syntax:
  - INSERT INTO tablename (column1,……,columnN) VALUES (val1,……..valN);
  - INSERT INTO tablename VALUES (val1,……..valN);
  - INSERT INTO tablename VALUES (val1,NULL,val3,NULL……..valN);
  - INSERT INTO tablename VALUES (val1,……..valN),(val1,……..valN);
- e.g.
  - INSERT INTO student (S_ID,name,address) VALUES (1,'Ram','Kathmandu');
  - INSERT INTO student VALUES (2,'Sita','Patan');
  - INSERT INTO student VALUES (3,NULL,'Patan');
  - INSERT INTO student VALUES (4,'Hari','Patan'), (5,'Gita','Pokhara');

Chapter 3 – Structured Query Language (SQL)

- **DML (Data Manipulation Language)**
- **To update data in table:**
- Syntax:
  - UPDATE tablename SET column1 = val1;
  - UPDATE tablename SET column1 = val1 WHERE column =val;
  - UPDATE tablename SET column1 = val1, column2 = val2…. WHERE column =val;
- e.g.
  - UPDATE student SET address = 'Dharan';
  - UPDATE student SET address = 'Dharan' WHERE S_ID = 2;
  - UPDATE student SET name='Sital', address = 'Birgunj' WHERE S_ID = 3;

Chapter 3 – Structured Query Language (SQL)

- **DML (Data Manipulation Language)**
- **To delete data in table:**
- Syntax:
  - DELETE FROM tablename ;
  - DELETE FROM tablename WHERE column =val;
- e.g.
  - DELETE FROM student;
  - DELETE FROM student WHERE S_ID = 5;

Chapter 3 – Structured Query Language (SQL)

- **DQL (Data Query Language)**
  - Query is an operation that retrieves data from one or more tables or views.
  - Query nested within another SQL statement is called a subquery.
  - SELECT is used in DQL.
  - Syntax:
    - SELECT * FROM tablename;
    - SELECT column1, column2 FROM tablename;
    - SELECT DISTINCT colum1, column2 FROM tablename;
    - SELECT * FROM tablename WHERE column operator value;
    - SELECT * FROM tablename WHERE column1 = value1 AND column2 = value2;
    - SELECT * FROM tablename WHERE column1 = value1 OR column2 = value2;
    - SELECT * FROM tablename WHERE column1 = value1 AND (column2 = value2 OR column3 = value3)
    - SELECT * FROM tablename ORDER BY cloumnname ASC|DESC;
    - SELECT TOP number|percent columnname(s) FROM tablename;

Chapter 3 – Structured Query Language (SQL)

- **DQL (Data Query Language)**
  - examples:
    - SELECT * FROM employee;
    - SELECT name, designation FROM employee;
    - SELECT DISTINCT address FROM tablename;
    - SELECT * FROM employee WHERE eid = 1;
    - SELECT designation, salary FROM employee WHERE eid = 1;
    - SELECT * FROM employee WHERE address = 'kathmandu' AND designation = 'manager';
    - SELECT * FROM employee WHERE address = 'kathmandu' OR designation = 'manager';
    - SELECT * FROM employee WHERE address = 'kathmandu' AND (designation = 'programmer' OR salary > 25000)
    - SELECT * FROM employee ORDER BY name ASC|DESC;
    - SELECT TOP 3 * FROM employee;(SqlServer)
    - SELECT * FROM employee LIMIT 3;(MySQL)

Chapter 3 – Structured Query Language (SQL)

- **DQL (Data Query Language)**
  - **Wildcard characters are used with SQL LIKE operator , to search for data within a table.**
  - **% => a substitute for zero or more characters**
  - **_ => a substitute for a single character.**
  - **[characterlist] => sets and range of characters to match.**
  - **[! characterlist] => match only character not specified in bracket.**
  - Syntax:
    - SELECT * FROM tablename WHERE columnname LIKE pattern;
    - e.g.
    - SELECT * FROM employee WHERE city LIKE '%s';
    - SELECT * FROM employee WHERE country LIKE '%land%';
    - SELECT * FROM employee WHERE country NOT LIKE '%land%';
    - SELECT * FROM employee WHERE city LIKE 'ber%';
    - SELECT * FROM employee WHERE city LIKE '_erlin';
    - SELECT * FROM employee WHERE city LIKE 'L_N_ON';
    - SELECT * FROM employee WHERE city LIKE '[bsp]%';
    - SELECT * FROM employee WHERE city LIKE '[a-d]%';
    - SELECT * FROM employee WHERE city NOT LIKE '[bsp]%';
    - SELECT * FROM employee WHERE city LIKE '[!bsp]%';

Chapter 3 – Structured Query Language (SQL)

- **DQL (Data Query Language)**
  - Syntax:
    - SELECT * FROM tablename WHERE columnname IN (val1,val2,….);
    - e.g
      - SELECT * FROM employee WHERE city IN ('london', 'paris');
    - SELECT * FROM tablename WHERE columnname BETWEEN value1 AND value2;
    - e.g.
      - SELECT * FROM employee WHERE salary BETWEEN 15000 AND 35000;
      - SELECT * FROM employee WHERE salary NOT BETWEEN 15000 AND 35000;
      - SELECT * FROM employee WHERE (salary BETWEEN 15000 AND 35000) AND NOT eid IN (1,2,3);
      - SELECT * FROM employee WHERE name BETWEEN 'A' AND 'M'
      - SELECT * FROM employee WHERE name NOT BETWEEN 'A' AND 'M'
    - SELECT columnname AS newname FROM tablename;
    - e.g.
      - SELECT designation AS post FROM employee;

Chapter 3 – Structured Query Language (SQL)

- **SQL functions:**
  - SQL has many built-in functions for performing calculations on data.
  - **AVG()** => returns average value of a numeric column.
    - *SELECT AVG(columnname) FROM tablename;*
  - **COUNT()** => returns the no. of rows.
    - *SELECT COUNT(columnname) FROM tablename;*
    - *SELECT COUNT(*) FROM tablename;*
  - **MAX()** => returns the largest value of selected column.
    - *SELECT MAX(columnname) FROM tablename;*
  - **MIN()** => returns the smallest value of selected column.
    - *SELECT MIN(columnname) FROM tablename;*
  - **SUM()** => returns the total sum of numeric column.
    - *SELECT SUM(columnname) FROM tablename;*

Chapter 3 – Structured Query Language (SQL)

- **SQL functions:**
  - **UPPER()** => converts value of field to uppercase.
    - *SELECT UPPER(columnname) FROM tablename;*
  - **LOWER()** => converts value of field to lowercase.
    - *SELECT LOWER(columnname) FROM tablename;*
  - **SQRT()** => used to find out square root of any number.
    - *SELECT SQRT(columnname) FROM tablename;*
  - **RAND()** => used to produce random numbers between 0 and 1.
    - *SELECT * FROM tablename ORDER BY RAND();*
  - **CONCAT()** => used to concatenate two strings to form single string
    - *SELECT CONCAT(col1,col2) FROM tablename;*

Chapter 3 – Structured Query Language (SQL)

- **GROUP BY and HAVING clauses:**
  - Used to divide the rows in table into groups.
  - Used with SQL aggregate functions like SUM(), AVG() etc.
  - e.g.
    - SELECT designation, SUM(salary) FROM employee GROUP BY designation
    - SELECT designation, MAX(salary) FROM employee GROUP BY designation HAVING MAX(salary) > 35000

Chapter 3 – Structured Query Language (SQL)

- **SQL using JOINS :**
- Is used to combine records from two or more tables in a database based on a common field between them.
- **Inner Join :**
- Select all rows from both tables as long as there is a match between the columns in both tables.
- **Syntax:**
  - SELECT columnname(s) FROM table1 INNER JOIN table2

    ON table1.column1 = table2.column2
- **e.g.:**
  - SELECT customer.name, order.oid FROM customer INNER JOIN order

    ON customer.cid = order.cid

Chapter 3 – Structured Query Language (SQL)

- **Left Join (left outer join):**
- Returns all rows from the left table (table1) with the matching rows in the right table (table2).
- The result is null in right side where there is no match.
- **Syntax:**
  - SELECT columnname(s) FROM table1 LEFT JOIN table2

    ON table1.column1 = table2.column2
- **e.g.:**
  - SELECT customer.name, order.oid FROM customer LEFT JOIN order

    ON customer.cid = order.cid

Chapter 3 – Structured Query Language (SQL)

- **RIGHT Join (right outer join):**
- Returns all rows from the right table (table2) with the matching rows in the left table (table1).
- The result is null in left side where there is no match.
- **Syntax:**
  - SELECT columnname(s) FROM table1 RIGHT JOIN table2
    ON table1.column1 = table2.column2
- **e.g.:**
  - SELECT customer.name, order.oid FROM customer RIGHT JOIN order
    ON customer.cid = order.oid

Chapter 3 – Structured Query Language (SQL)

- **Full Outer Join:**
- Returns all rows from the left table (table1) and from right table (tabl21).
- Combines the result of both left and right join.
- **Syntax:**
  - SELECT columnname(s) FROM table1 FULL OUTER JOIN table2

    ON table1.column1 = table2.column2
- **e.g.:**
  - SELECT customer.name, order.oid FROM customer FULL OUTER JOIN order

    ON customer.cid = order.oid

Chapter 3 – Structured Query Language (SQL)

- **Subquery:**
  - Is a query within another SQL query and embedded within where clause.
  - Is used to return data that will be used in main query as a condition to further restrict the data to be retrieved.
  - Can be used with select, insert, update and delete statements along with operators like =,>,<,>=,<=, IN, BETWEEN etc.
  - **Rules for subquery:**
    - Must be enclosed with in parenthesis.
    - Can have only one column in SELECT clause.
    - An 'ORDER BY' can not be used, but 'GROUP BY ' can be.
    - Subquery that return more than one row can only be used with multiple value operator like IN operator.
    - A subquery can't be immediately enclosed in a set function.
    - 'BETWEN' operator can't be used with subquery, however 'BETWEEN' operator can be used within subquery.
    - SELECT list cannot include any references to values that evaluate BLOB,ARRAY.

Chapter 3 – Structured Query Language (SQL)

- **Subquery with SELECT statement**
- e.g.:
  - SELECT * FROM employee WHERE eid IN (SELECT eid FROM employee WHERE salary > 30000)
- **Subquery with INSERT statement**
- e.g.:
  - INSERT INTO employee1 SELECT * FROM employee WHERE eid IN (SELECT eid FROM employee)
- **Subquery with UPDATE statement**
- e.g.:
  - UPDATE employee SET salary = salary * 0.25 WHERE designation IN (SELECT designation FROM employee WHERE designation LIKE 'd%')
- **Subquery with DELETE statement**
- e.g.:
  - DELETE FROM employee WHERE salary IN (SELECT salary FROM employee WHERE salary > 30000)

Chapter 3 – Structured Query Language (SQL)

- **Set Operations:**
  - SQL used set operations to combine same type of data from two or more tables.
  - Set operations are based on principle of mathematical set theory.
  - To perform Set operations, **Union Compatibility** property must be hold i.e. the relations involved in the operations must have *an equal no. of attributes* and they must have *the same domain of attributes*.
  - Different set operations are:
    - UNION
    - INTERSECT
    - EXCEPT

Chapter 3 – Structured Query Language (SQL)

- **Set Operations:**
  - **UNION**
    - Combines two or more result sets into a single set, without duplicates.
    - Any duplicate records are automatically removed unless **UNION ALL** is used.
    - e.g.
      - SELECT name, address FROM student

        UNION

        SELECT name, address FROM teacher
  - **INTERSECT**
    - Takes the data from both result sets which are in common.
    - Removes duplicate rows from the final result set.
    - **INTERSECT ALL** operator does not remove duplicate rows from the final result set.

Chapter 3 – Structured Query Language (SQL)

- **Set Operations:**
  - **INTERSECT**
    - e.g.
      - SELECT name, address FROM student

        INTERSECT

        SELECT name, address FROM teacher
  - **EXCEPT**
    - Takes the data from first result set, but not the second.
    - Automatically eliminates duplicates.
    - **EXCEPT ALL** retains all duplicates.
    - e.g.
      - SELECT name, address FROM student

        EXCEPT

        SELECT name, address FROM teacher

Chapter 3 – Structured Query Language (SQL)

- **Views**
  - View is a virtual table based on the result set of an SQL statement
  - Is the logical representation of data.
  - It contains rows and columns just like real table.
  - SQL functions, where and join statements can be added to a view and data can be presented as if data are coming from one single table.
  - Views allow users to
    - Structure data in a way that users want.
    - Restrict access to data such that user can see and sometimes modify exactly what they need and no more
    - Summarize data from various tables which can be used to generate reports.
  - **Syntax to create view**
    - CREATE VIEW viewname AS

      SELECT columnname(s) FROM table; / WHERE clause

Chapter 3 – Structured Query Language (SQL)

- **Views**
  - **e.g. creating view**
    - CREATE VIEW view_customer AS

      SELECT name, phone FROM customer WHERE address = 'kathmandu'
  - **Syntax to query view**
    - SELECT * FROM viewname
  - **e.g. to query view**
    - SELECT * FROM view_customer
  - **Syntax to update view**
    - UPDATE viewname SET column = value WHERE clause
    - ***It should not contain aggregate function or distinct or group by clause while updating views.***

Chapter 3 – Structured Query Language (SQL)

- **Views**
  - **e.g. to update view**
    - UPDATE view_customer SET phone = 102050 WHERE name = 'ram'
  - **Syntax to drop view**
    - DROP VIEW viewname
  - **e.g. to drop view**
    - DROP VIEW view_customer

Chapter 3 – Structured Query Language (SQL)

- **STORED PROCEDURE**
- Stored procedure is a set of SQL statements with an assigned name, which are stored in RDBMS as a group, so it can be reused and shared by multiple programs.
- We can also pass parameters to a stored procedure, so that stored procedure can act based on the parameter values(s) passed.
- **Syntax:**
  - CREATE PROCEDURE procedurename
    AS
    SQL statements
    GO;
  - EXEC procedurename
- **Example:**
  - CREATE PROCEDURE selectemp
    AS
    SELECT * FROM employee
    GO
  - EXEC selectemp

Chapter 3 – Structured Query Language (SQL)

- Stored procedure with parameters
  - CREATE PROCEDURE selectemp @job varchar(30)
    AS
    SELECT * FROM employee WHERE job = @job
    GO;
  - EXEC selectemp @job = 'programmer'

  - CREATE PROCEDURE selectemp @job varchar(30) , @salary int
    AS
    SELECT * FROM employee WHERE job = @job AND salary > @salary
    GO;
  - EXEC selectemp @job = 'programmer' , @salary > 25000

Chapter 3 – Structured Query Language (SQL)

- QBE (Query By Example) is developed at IBM in early 1970's.
- It has two dimensional syntax, and queries looks like table.
- QBE queries are expressed by examples.
- QBE queries are expressed by using skeleton tables, which show the relational schema of the database.
- Skeleton table looks like

Relation name	Column1	Column2	…… ColumnN
Tuple or row operation			

- For relation college (collegeid, name), QBE is as shown below

college	collegeid	name

Chapter 3 – Structured Query Language (SQL)

- Use of P to get list of relation.
- P. for print command.
- Data retrieval command is P. <variable name> where variable name is optional.
- QBE automatically eliminates duplicate values.
- To see the duplicate value, P.ALL
- QBE supported by Microsoft Access is Graphical QBE

Chapter 3 – Structured Query Language (SQL)

- **Example**
- employee (name, city, salary)
  - To find all employee names who live in city 'kathmandu'

employee	name	city	salary
	P. or P.ALL	kathmandu	

  - To find entire employee relation

employee	name	city	salary
P.			

  - To find employee name whose salary is greater than 15000

employee	name	city	salary
	P.		> 15000

  - To find all employees whose salary between 10000 and 25000

employee	name	city	salary
P.			>= 10000 AND <= 25000

Chapter 3 – Structured Query Language (SQL)

- **Database Modification in QBE**
  - To INSERT

employee	name	city	salary
I.	Ram	Pokhara	25000

  - To DELETE

employee	name	city	salary
D.	Ram		

    - To delete city of Ram

employee	name	city	salary
	Ram	D.	

  - To UPDATE

employee	name	city	salary
	Ram		U.35000