BABASAHEB BHIMRAO AMBEDKAR UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

LECTURE TOPIC - SQL (Structured Query Language)

PAPER NAME - ADVANCE DBMS
SEMESTER - FOURTH
FACULTY - SUGANDHA SINGH

1. Introduction:

SQL language provides a high level de larative language interface, so the user only specifies what the result is to be, leaving the actual oprimization and decisions on how to execute the query to the DBMS. SQL was implemented and designed at IBM research. SQL is a comprehensive database language.

1.1 Attribute Data Types and Domains in SQL

- Numeric data types include integer no.s of various sizes (INTEGER or INT and SMALLINT) and floating point no.s of various precision(FLOAT or REAL and DOUBLE PRECISION)
- Character string data types are either fixed length - CHAR(n) or CHARACTER(n), where n is a no. of characters or varying length - VARCHAR(n)
- Bit-String data types are either of fixed length n - BIT(n) or varying length - BIT VARYING(n), where n is the maximum no. of bits.
- DATE data type has ten positions, and its components are YEAR, MONTH , and DAY in the form YYYY-MM-DD.
- TIME data types has atleast eight positions, with the components HOURS, MINUTE, and SECOND in the form HH:MM:SS

2. Types of SQL

2.1 DDL(Data definition language) Commands

CREATE, ALTER, DROP, GRANT, REVOKE, TRUNCATE AND COMMENT

2.2 DML (Data manipulation language) Commands

SELECT, INSERT, UPDATE, DELETE, CALL, LOCK TABLE AND EXPLAIN PLAN

2.1 DATA DEFINITION LANGUAGE

2.1.1 CREATE COMMAND

CREATE COMMAND can be used to create Schemas, Tables, Views, domains, assertions and triggers.

a) CREATE SCHEMA COMMAND

A schema is created via the **CREATE SCHEMA** statement, Which can includes all the schema elements definition.
For example, the following statement creates a schema called COMPANY, owned by the user with authorization identifier 'Ram'.

CREATE SCHEMA COMPANY AUTHORIZATION Ram;

In general not all users are authorized to create schemas and schema elements. The privilege to create schemas, tables, and other constructs must be explicitly granted to the relevant user accounts by the system administrator or DBA.

b) CREATE TABLE COMMAND in SQL

The ***CREATE TABLE*** command is used to specify a new relation by giving it a name and specifying its attributes and initial constraints.
The attribute is specified first and each attribute is given a name, a data type to specify its domain of values and any attribute constraints, such as NOT NULL.
Typically, the SQL schema in which the relations are declared is implicitly specified in the environment in which the CREATE TABLE statements are executed. Alternatively we can explicitly attach the schema name to the relation name, separated by a period. For example, by writing
CREAT TABLE COMPANY.EMPLOYEE…
rather than
CREATE TABLE EMPLOYEE…

IN Figure 1, we can explicitly (rather than implicitly) make the EMPLOYEE table part of the COMPANY schema.

CREATE TABLE EMPLOYEE
(**Firstname** VARCHAR(15) NOT NULL,
MiddleName CHAR,
LastName VARHAR(15) NOT NULL,
Ssn CHAR(9) NOT NULL,
Dob DATE,
Address VARCHAR(30),
Gender CHAR,
Salary DECIMAL(10,2),
Super_ssn CHAR(9),

Dept_no INT NOT NULL
PRIMARY KEY (Ssn),
FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn));

CREATE TABLE DEPARTMENT
(**Dept_name** VARCHAR(20) NOT NULL,
Dno INT NOT NULL,
Mgr_ssn CHAR(9),
Mgr_start_date DATE,
PRIMARY KEY(Dno),
UNIQUE(Dname),
FOREIGN KEY(Mgr_ssn) REFERENCES EMPLOYEE(Ssn));
Figure 1

c) CREATE DOMAIN COMMAND

We can create a domain SSN by the following statement:

CREATE DOMAIN SSN AS CHAR(9);
we can use SSN in place of CHAR (9)

The **CHECK** clause can also be used in conjunction with the CREATE DOMAIN statement.
For example, we can write the following statement:
CREATE DOMAIN D_No AS INTEGER CHECK (D_No >0 AND D_No<15);

We can then use the created domain D_No as the attribute type for all attributes that refer to department numbersin Figure 1, such as Dept_no of EMPLOYEE and dno of DEPARTMENT.

d) The DROP Command

The DROP command can be used to drop schema and schema elements such as tables, domains or constraints.
There are two types of drop behavior options: CASCADE and RESTRICT
For example, to remove the COMPANY database schema and all its tables, domains and other elements, the CASCADE opttion can be used as:

DROP SCHEMA COMPANY CASCADE;

If the RESTRICT option is used in place of CASCADE, the schema is dropped only if it has no elements in it, otherwise the DROP command will not be executed.

If we want to drop relation only not schema use following command:

DROP TABLE EMPLOYEE CASCADE;

If the RESTRICT option is chosen instead of CASCADE, a table is dropped only if it ia not referenced in any constraints (for example, foreign key or view)

e) The ALTER Command

The ALTER Command can be used to change or modify schema element or base table definitions. For base tables, rhe ALTER TABLE include actions : adding or dropping a column(attributes), changing a column definitions and adding or dropping table constraints.
To add new attribute to the EMPLOYEE base relations in the COMPANY schema we can use the command:

ALTER TABLE COMPANY.EMPLOYEE ADD COLUMN Designation VARCHAR(15);

To drop a column, we can use either RESTRICT or CASCADE action.
ALTER TABLE COMPANY.EMPLOYEE DROP COLUMN DOB CASCADE;

If **CASCADE** is used, all constraints and views that reference the column are dropped automatically from the schema along with column.
The RESTRICT Command can be executed successfull oly if no views or constraints are reference the column.
We can also alter a column definition by dropping an existing default clause or by defining a new default clause. For example,

ALTER TABLE COMPANY.EMPLOYEE ALTER COLUMN Super_ ssn DROP DEFAULT;
ALTER TABLE COMPANY.EMPLOYEE ALTER COLUMN Super_ ssn SET DEFAULT '226677' DEFAULT;

2.2 Data Manipulation Commands

a) The INSERT command

The INSERT command is used to insert add a tuple to a relation. We must specify the relation name and a list of values for the tuple. The values should be listed in thw same order in which the corresponding attributes were specified in the CREATE TABLE command.
Syntax: INSERT INTO table_name VALUES (value1,value2,.,);

or

Syntax: INSERT INTO table_name (column1, column2,...) VALUES (value1,value2,.,);

For example, to add a new tuple tuple in EMPLOYEE relation we can use:

INSERT INTO EMPLOYEE VALUES ('Rajan','kumar','Dev','332244','1990-06-28','21 Dev nagar, Gwalior, MP','M',45000,'622311',2);

b) The DELETE command

The **DELETE** command removes the tuples in a relation. It can also used with **WHERE** clause to select the tuples to be deleted. If **WHERE** clause is not used with the statement then all tuples will be deleleled.
Synax: DELETE FROM table_name WHERE some_column=some_value;

For example, to delete all tuples from **EMPLOYEE** relation:

DELETE FROM EMPLOYEE;

To delete a selected tuples from EMPLOYEE relation we must used WHERE clause to specify condition. For example, to delete zero, one or more tuples following statement can be used:

DELETE FR0M EMPLOYEE WHERE Fname='Ramesh';
DELETE FR0M EMPLOYEE WHERE Ssn='331188';

c) The SELECT statement

THE SELECT statement is used fir retrieving informarion from a database. The SELECT statement defines WHAT is to be returned. The FROM statement defines the table(s) or View(s) used by SELECT or WHERE statements. "*" means all columns will be return from tables.
Syntax: SELECT * FROM table_name;
The WHERE clause define what records are to be included in the query, it is optional.
Syntax: SELECT <attribute list> FROM <table list> WHERE <condition>;
The WHERE clause uses conditional operators:

1) =,>,>=,<,<=,!=
2) BETWEEN x AND y
3) IN
4) LIKE '%string'("%" is a wild card)
5) IS NULL
6) NOT (BETWEEN/IN/LIKE/NULL)

The SELECT DISTINCT statement is used to return only distinct(different) values. In a table, a column may contain many duplicate values; and sometimes we want to list the different (distinct) values.
Syntax: SELECT DISTINCT column_name, column_name FROM table_name;

Query1: Retrieve Salary, Address FROM EMPLOYEE where Fname='Suresh' AND Mname='kumar' AND Lname='Sachdev';
This query will return the record from a table EMPLOYEE that satisfy the condition of WHERE clause.

Salary	Address
40000	72, Ring Road, Indore, MP

Query2: Retrieve the name and address of all employees who work for research department.
SELECT Fname,Mname,Lname,Address **FROM** EMPLOYEE,DEPARTMENT **WHERE** Dept_name='Research' **AND** Dno=Dept_no;
Query2 is similar to SELECT-PROJECT-JOIN sequence of relational algebra operations, Such queries are called select-project-join queries. In the WHERE clause of Query2 , the condition Dept_name='Research' is a selection condition.
The ORDER BY statement can be used to defines how the records are to be sorted. Default is to order in ASC (Ascending) order. To sort the records in a descending order, use DESC keyword.
Syntax: SELECT column_name,column_name FROM table_name ORDER BY column_name,column ASC/DESC;

d) **The UPDATE statement**
The UPDATE statement is used to modify attribute value or records in a table. A WHERE clause in the UPDATE command selects the tuples to be modified from a relation. An additional SET clause in the UPDATE command specifies the atributes to be modified and their new values.
Syntax: UPDATE table_name SET column1=value1,column2=value2,... WHERE some_column=some_value;

For example, to change the address and salary of employee whose department number is 2.
UPDATE EMPLOYEE SET Address='91 kesav nagar, MP', Salary='30000' WHERE Dept_no=2;