Date .....

Chapter-6 Chapter-7

Basic SQL.

More Complex SQL Retrieval Queries

* SQL
  → SQL - Structured Query language.
  → Originally, called SEQUEL (Structured English Query language)
  → High-level declarative language user specifies what result is to be.

* Schema in SQL (Not used in MySQL)
  → Schema was introduced in order to group together tables and other constructs.
  → SQL schema is identified by a schema name, and an authorization identifier to indicate the user or account, as well as descriptors for each element in the schema.
  * CREATE SCHEMA COMPANY AUTHORIZATION 'James'
    (name of schema; 'James' = owner)

* Creating a table in SQL.
  → CREATE TABLE = Specifies new relation by giving names and data types to its attributes

```
* CREATE TABLE STUDENT
    ( RollNo  Varchar(6)  NOT NULL,
      Name    Varchar(15) NOT NULL,
      DOB     Date );
```

* Constraints in SQL
  → Constraints are used to limit the type of data to be added. These are rules enforced on the data columns.

Date .....

→ Various constraints like :

- NOT NULL - Ensures that a column cannot have a null value.
- DEFAULT - Provides a default value for a column when none is specified.
- UNIQUE - Ensures that all values are different in a column.
- PRIMARY KEY - Uniquely identifies each row / ~~column~~ record in database table.
- FOREIGN KEY - Uniquely identifies a row in any given database table.
- CHECK - Ensures that all the values in a column satisfies certain conditions.
- INDEX - Used to create and retrieve data from the database very quickly.

Check - not handled in mysql

* Keys

→ Primary Key

A primary key is a set of one or more attributes that can uniquely identify tuples within the relation.

→ Candidate Key

All attribute combinations inside a relation that can serve as primary key are Candidate Keys as they are candidate for the primary key position.

→ Alternate Key

A candidate key that is not the primary key is called an alternate key.

Date .....

→ Foreign Key
A non-key attribute, whose values are derived from the primary key of some other table, is known as Foreign-Key in its current table.

* Drop Table
→ Drop table is used to remove a relation (base table) and its definition
→ The relation can no longer be used in queries, updates, or any other commands since its description no longer exists.

✡ DROP TABLE STUDENT ;

→ Two options to delete a table :
• The CASCADE option allows you to remove the table and its dependent objects. This drops the table and any other constraint or views which are referencing this table forcibly and automatically.

✡ DROP TABLE STUDENT CASCADE ;

• The RESTRICT option rejects the removal if there is any object depends on the table. Restrict option is default if you don't explicitly specify it in the DROP statement. Restrict means delete and remove its definition iff the table is not being referenced in any other table.

✡ DROP TABLE STUDENT RESTRICT ;

Date .....

* Alter Table

→ The ALTER TABLE statement is used to add, delete or modify columns in an existing table.

→ It is also used to add and drop various constraints.

→ The new attribute will have NULLs in all the tuples of the relation right after the command is executed, hence the NOT NULL constraint is not allowed for such an attribute.

* ALTER TABLE EMPLOYEE ADD JOB VARCHAR(12);

* ALTER TABLE DEPARTMENT
ADD CONSTRAINT FK1 FOREIGN KEY (MGRSSN)
REFERENCES EMP (SSN);

* Insert, Delete and Update Statements in SQL.

→ There are 3 SQL commands to modify the database : INSERT, DELETE and UPDATE.

INSERT

→ In its simplest form, it is used to add one or more tuples to a relation.

→ Values should be listed in the same order as the attributes were specified in CREATE TABLE command.

→ Or attributes should be specified with INSERT in order.

→ Only the constraints specified in the DDL command are automatically enforced by the DBMS when updates are applied to the db.

→ Another variation of INSERT allows insertion of multiple tuples resulting from a query into a relation

Date .....

* INSERT INTO STUDENT
  VALUES ('AC-1201', 'Abhishek', 25-01-2001);

DELETE
→ Delete clause removes tuples from a relation.
→ 'Where' clause specifies a selected tuple to be deleted and if it is missing then all tuples in relation are deleted.
→ Tuples are deleted from only one table. No. of tuples to be deleted depends on no. of tuples that satisfy in Where clause.
→ Referential integrity should be enforced.

* DELETE FROM EMPLOYEE
  WHERE LNAME = 'Brown';

UPDATE
→ Used to modify attribute values of one or more selected tuples.
→ 'Where' clause selects the tuple to be ~~deleted~~ modified.
→ SET clause Specifies the attributes to be modified and their new values.
→ Referential integrity should be enforced.

* UPDATE PROJECT
  SET PLOCATION = 'Bellaire', DNUM = 5
  WHERE PNUMBER = 10;

multi-set ⇒ multiple sets

Date .....

* Retrieval Queries in SQL.

→ SELECT statement is used to retreive information from a database

→ An SQL relation (table) is a multi-set (sometimes called a bag) of tuples; it is not a set of tuples. Justification: A set has no duplicate values whereas a relational model allows to have more tuples with identical attributes. Hence, it called a multi-set that is multiple sets.

→ SQL relations can be constrained as sets by specifying primary key or UNIQUE constraint. or by using DISTINCT in queries.

* SELECT <attribute list>
FROM <table list>
WHERE <condition>

→ The SELECT clause specifies the projection attributes and WHERE clause specifies the selection condition.

* Aliases

→ Aliases are temporary labels given along with the relation name.

* SELECT E.FNAME, E.LNAME, S.FNAME
FROM EMPLOYEE E, EMPLOYEE S
WHERE E.SUPERSSN = S.SSN

Here, E and S are aliases or tuple variables.

Date .....

* Taking input from user while executing a query.
→ User input through = #substitution.

* SELECT ename from EMPLOYEE
WHERE DNO = #a;

* Use of *
→ To retrieve all the attributes values of selected tuples, a * is used, this stands for 'all the attributes'.

* Use of DISTINCT
→ A relation is not treated as a set in SQL, so duplicate tuples can appear.
→ For the elimination of duplicate tuples in a query result, the keyword DISTINCT is used.

* SELECT DISTINCT Salary
FROM EMPLOYEE;

* Nesting of Queries.
→ A complete SELECT query, called a nested query can be specified within the WHERE clause of another query, called the outer query.

* SELECT FNAME, LNAME, ADDRESS
FROM EMPLOYEE
WHERE DNO IN (SELECT DNUMBER FROM DEPARTMENT
WHERE DNAME = 'Research');

Date .....

* Correlated Nested Queries.

→ If a condition in the WHERE clause of a nested query references an attribute of a relation declared in outer query, the 2 queries are said to be correlated.

→ The result of a correlated nested query is different for each tuple of relations the outer query.

→ A query written with nested SELECT ... FROM ... WHERE .... blocks and using the = or IN comparison operators can always be expressed as a single block query.

* Set Operations

→ Some set operations are incorporated in SQL
  i) Union operation (UNION)
  ii) set difference (MINUS)
  iii) intersection (INTERSECT)

→ Resulting relation of these set operations are set of tuples; duplicate tuples are eliminated from the result.

→ Set operations apply only to union compatible relations, two relations must have same attributes and attributes must appear in the same order

```
SELECT column_names FROM table1
UNION
SELECT column_names FROM table2;
```

* EXISTS Function

→ EXISTS Function is used to check whether the result of a correlated nested query is empty or not.

Date .....

* Explicit Sets
→ It is possible to use an explicit (enum) set of values in the WHERE clause rather than a nested query.

* SELECT DISTINCT SSN
FROM WORKS_ON
WHERE PNO IN (1,2,3)

* NULL in SQL Queries
→ We can check if a value is NULL.
→ IS or IS NOT is used to compare NULLs bcoz it considers each NULL distinct from other.
→ If a join condition is specified, tuples with NULL values for the join attributes are not included in the result.
→ NULL represents
i) Unknown value
ii) Unavailable or withheld value
iii) Not applicable attribute

* Aggregate Functions
→ These include COUNT, SUM, MAX, MIN, AVG

* Join
→ A join is a query that combines rows from 2 or more tuples.
→ In a join-query, more than 1 table are listed in FROM clause.
→ Function of combining data from multiple tables is called joining.

Date .....

* SELECT Fname, Lname, Address
FROM EMPLOYEE, DEPARTMENT
WHERE Dname = 'Research' AND DNumber = DNO;

→ Cartesian Product is known as unrestricted join. In this 2 tables, all possible concatenations are performed of all rows of both the tables.
→ Allows the user to specify different types of joins (regular "theta" JOIN, NATURAL JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, CROSS JOIN, etc).
→ Equi-Join
The join, in which columns are compared for equality, is called Equi-Join.
→ Natural Join
The join, in which only one of the identical columns (coming from joined tables) exists, is called Natural-Join.

* SELECT FNAME, LNAME, ADDRESS
FROM (EMPLOYEE NATURAL JOIN DEPARTMENT
AS DEPT(DNAME, DNO, MSSN, MSDATE)
WHERE DNAME = 'Research';

* Grouping
→ Each subgroup of tuples consists of the set of tuples that have the same value for the grouping attributes.
→ GROUP BY - clause is used to specify the groups attributes, which must appear in the SELECT clause.
→ GROUP BY applies the aggregate functions independently to a series of groups that are defined by having a field value in common.

Date .....

```
SELECT  DNO, COUNT(*), AVG(SALARY)
FROM    EMPLOYEE
GROUP BY DNO;
```

* HAVING - Clause
→ HAVING clause is used for specifying a selection condion on groups.

```
SELECT  PNUMBER, PNAME, COUNT(*)
FROM    PROJECT, WORKS_ON
WHERE   PNUMBER = PNO
GROUP BY PNUMBER, PNAME
HAVING  COUNT(*) > 2;
```

* Substring Comparison
→ LIKE comparison operator is used to compare partial strings.
→ % and _ are used.
→ % replaces an arbitrary no. of characters
→ _ replaces a single arbitrary character.

```
SELECT  Fname, Lname
FROM    Employee
WHERE   Address like '% Houston %';
```

```
SELECT  Fname, Lname
FROM    Employee
WHERE   Lname like 'S___th';
```

* Arithmetic Operations
→ Standard arithmetic operators +, -, *, / can be applied to numeric values in SQL query result.

Date .....

* Order by
→ The ORDER BY clause is used to sort the tuples in a query result based on the values of some attributes.
→ Default order is ascending.
→ If DESC is specified, the order is descending. ASC can also be explicitly specified for ascending order.

* SELECT DName, LName, FName
FROM DEPARTMENT, EMPLOYEE, WORKS_ON, PROJECT
WHERE DNumber = DNO AND SSN = ESSN
AND PNO = PNUMBER
ORDER BY DName, LName;

* Constraints as Assertion.
→ Constraints that do not fit in the basic SQL categories
→ To cover this SQL supports the creation of assertions that are constraints not associated with only one table.

* CREATE ASSERTION [assertion_name]
CHECK ([condition]);

* SQL Triggers.
→ Triggers are made to monitor a database and take action when a condition occurs.
→ Triggers are expressed in a syntax similar to assertions and include the following:
- event (eg. update)
- condition
- action (to be taken when condition is satisfied).

Date .....

* Views in SQL.

→ View is a virtual table that is derived from other tables.
→ Allows limited update operations
→ Allows full query operations.
→ A convenience for expressing certain operations
→ View is supposed to be always up-to-date.
→ The view does not have to be realized or materialized at the time of view definition but rather at the time when a query is specified on the view.
→ Two main approaches used to keep view up to dat
- Query modification
- View materialization

* CREATE VIEW WORKS_ON1
AS SELECT Fname, Lname, Pname, Hours
FROM Employee, Project, Works_On
WHERE SSN = ESSN AND Pno = Pnumber;

- Query Modification

Query modification involves modifying or transformi the view query into a query on the underlying base tables.

- View Materialization

Keeps permanent copy of view.
Use incremental update using one of the strategies
i) immediate update strategy updates a view as soon as base table chang
ii) lazy update strategy updates a view when needed by a view query
iii) periodic update strategy updates the view periodically.

Spiral

Date .....

* Insertion, deletion, updates through view
→ An update on a view defined on a single table without any aggregate functions can be mapped to an update on the underlying base table under certain conditions like view must have primary key, only simple attributes, aggregate functions.
→ In case view involves joins, an update operation may be mapped to update operations on the underlying base relations in multiple ways.

* Views as Authorization Mechanisms.
→ Views are also used for database security and authorization mechanisms by hiding certain attributes or tuples from unauthorized users.

* Altering Table.

* ALTER TABLE Persons
ADD Constraint chk_Person CHECK (P_Id > 0 and City = 'Sandnes');