Database

A collection of related data.

<u>Database Applications</u>

- Banking: all transactions
- Airlines: reservations, schedules
- Universities: registration, grades
- Sales: customers, products, purchases
- Manufacturing: production, inventory, orders, supply chain
- Human resources: employee records, salaries, tax deductions

Types of Database

1. Internet Database
2. Multimedia Database
3. Mobile Database
4. Clustering Based Disaster proof Database
5. Geographic Information System
6. Genome Data Management

Logical Database Design of the databases

- Building a Database
  - Designing the Database
    - (Think before your create)
  - Creating a new Database
    - (Name and location only)
  - Defining the Database structure
    - (Schema and data)
  - Entering data
    - (Loading or automation)
  - Define additional properties
    - (Validation, relationships, networks)

ER Diagram

- Entity Relationship Diagrams (ERDs) illustrate the logical structure of databases.

Entity Relationship Diagram Notations

Entity:

An entity is an object or concept about which you want to store information

Key attribute:

A key attribute is the unique, distinguishing characteristic of the entity. For example, an student's ID might be the Student's key attribute.

Multivalued attribute:

A multivalued attribute can have more than one value. For example, an employee entity can have multiple skill values.

Derived attribute:

A derived attribute is based on another attribute. For example, an Student's Age is based on the his birthdate.

Relationships:

Relationships illustrate how two entities share information in the database structure.

Cardinality Ratio :

Cardinality ratio between Entity1 and Entity2 is 1:N

SQL

Structured **Q**uery **L**anguage

SQL is a Standard

- SQL is an ANSI (American National Standards Institute) standard .
- SQL is a Computer language for accessing and manipulating database systems.
- SQL statements are used to retrieve and update data in a database.
- SQL works with database programs like MS Access, DB2, Informix, MS SQL Server, Oracle, Postgres, Sybase, etc.

SQL Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. “Student" or “Department"). Tables contain records (rows) with data.
Below is an example of a table called “Student":

StudentID	StudentName	Address	BirthDate
1024	Ravi	23 hauz khas, New Delhi	08/11/1986
1025	Kisore	Malviya nagar, New Delhi	1/8/1988
1026	krishna	Vasant Vihar, New Delhi	4/5/1987

SQL Data Definition Language (DDL)

SQL Data Definition Language (DDL) is used to create and destroy databases and database objects. These commands will primarily be used by database administrators during the setup and removal phases of a database project.

- **CREATE** - to create and manage many independent databases object.
- **ALTER** - to make changes to the structure of a table without deleting and recreating it
- **DROP** - remove entire database objects

Create Table

```
CREATE TABLE Student
 ( StudentNo integer Primary key,
   StudentName  varchar(30),
   Address varchar(200),
   Birthdate date,
   Sex char(1),
   DNo integer,
   HostelID integer
)
CREATE TABLE Department
( DNumber integer primary key,
 DName varchar(30)
)
```

Data Type

Integer: Number Type
Date : calendar date (year, month, day)
Varchar (): variable-length character string
Char (): fixed-length character string
Boolean: logical Boolean (true/false)
Numeric [(*p*, *s*)]: exact numeric of selectable precision

Drop Database

DROP DATABASE University;

Drop Table

Drop Table Student

SQL Queries

With SQL, we can query a database and have a result set returned.
A query like this:

```
SELECT StudentName FROM Student
```

Gives a result set like this:

StudentName
Ravi
Kishore
Krishna

SQL Data Manipulation Language (DML)

SQL (Structured Query Language) is a syntax for executing queries. But the SQL language also includes a syntax to update, insert, and delete records.
These query and update commands together form the Data Manipulation Language (DML) part of SQL:

- **SELECT** - extracts data from a database table
- **UPDATE** - updates data in a database table
- **DELETE** - deletes data from a database table
- **INSERT INTO** - inserts new data into a database table

SQL The SELECT Statement

The SELECT statement is used to select data from a table. The tabular result is stored in a result table (called the result-set).

Syntax

```
SELECT column_name(s)
FROM table_name
WHERE  condition
```

Select All Columns: use a * symbol instead of column names, like this:

Query 1: Gives the all faculty name**.**

```
SELECT   *
FROM     FACULTY;
```

Query 2: Gives the details of faculties.

```
SELECT   FACULTYNAME, DESIGNATION,  SALARY
FROM     FACULTY;
```

Query 3: Gives the details Course offered by university

```
SELECT   COURSENO, NUMCREDIT
FROM     COURSE;
```

Semicolon after SQL Statements?

Semicolon is the standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.

Some SQL tutorials end each SQL statement with a semicolon. Is this necessary? In MS Access and SQL Server we do not have to put a semicolon after each SQL statement, but some database programs force you to use it.

Where Condition

The WHERE clause is used to specify a selection criterion.

The WHERE Clause

To conditionally select data from a table, a WHERE clause can be added to the SELECT statement.

Syntax

SELECT column FROM table

WHERE column operator value

With the WHERE clause, the following operators can be used:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern

Note: In some versions of SQL the <> operator may be written as !=

Using the WHERE Clause

Query 5: Find out the male student.

```
SELECT   STUDENTNAME, SEX
FROM     STUDENT
WHERE    SEX = 'M';
```

Query 7: Gives the name of student who are hostler.

```
SELECT   STUDENTNAME, CITY
FROM     STUDENT
WHERE    HOSTELID < > NULL
```

Using Quotes

Note that we have used single quotes around the conditional values in the examples.

SQL uses single quotes around text values (most database systems will also accept double quotes). Numeric values should not be enclosed in quotes.

For text values:

This is correct:

SELECT * FROM Student WHERE StudentName=‘Ravi'

This is wrong:

SELECT * FROM Student WHERE StudentName= Ravi

SQL The INSERT INTO Statement

The INSERT INTO Statement

The INSERT INTO statement is used to insert new rows into a table.

Syntax

INSERT INTO table_name

VALUES (value1, value2,....)

You can also specify the columns for which you want to insert data:

INSERT INTO table_name (column1, column2,...)

VALUES (value1, value2,....)

Insert a New Row

And this SQL statement:

```
INSERT INTO Student
VALUES (1029, ‘harish', ‘Dwarka New Delhi', 08/7/1987)
```

SQL The UPDATE Statement

The Update Statement

The UPDATE statement is used to modify the data in a table.

Syntax

UPDATE table_name

SET column_name = new_value

WHERE column_name = some_value

Update one Column in a Row

We want to add a first name to the student table to ‘Nina’ whose StudentID is 1029

```
UPDATE Student SET StudentName = 'Nina'
WHERE StudentID = 1029
```

SQL The Delete Statement

The Delete Statement

The DELETE statement is used to delete rows in a table.

Syntax

DELETE FROM table_name

WHERE column_name = some_value

"Nina" is going to be deleted:

DELETE FROM Student WHERE StudentID = 1029

Delete All Rows

It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

DELETE FROM table_name

Or

DELETE * FROM table_name

Avg, Max, Min, Count

Query 11: Gives the number of student in university

```
SELECT    COUNT (*)
FROM      STUDENT;
```

Query 12: Maximum salary of faculty.

```
SELECT    MAX (SALARY)
FROM      FACULTY;
```

Query 13: Gives the average salary of faculty whose salary are more 40,000.

```
SELECT    AVG (SALARY) AS 'Average Salary'
FROM      FACULY
WHERE     SALARY >40000;
```

Join

Dynamically relate different tables by applying what's known as a join

Query 15:
Find the names of student who are in COURSENO CEL747, and got GRADE =F;

SELECT STUDENTNAME
FROM STUDENT s **INNER JOIN** ENROLL e **ON** e.STUDENTNO = s.STUDENTNO
WHERE COURSENO = ‘CEL747’ **AND** GRADE = ‘F’;

ORDER BY Clause

Query 8:

Find all three credit course and list them alphabetically by coursename in descending order.

```
SELECT COURSENAME
FROM COURSE
WHERE NUMCREDIT = 3
ORDER BY COURSENAME DESC;
```