**Relational model in DBMS**

In relational model, the data and relationships are represented by collection of inter-related tables. Each table is a group of column and rows, where column represents attribute of an entity and rows represents records.

**Sample relationship Model**: Student table with 3 columns and four records.

**Table: Student**

|  |  |  |
| --- | --- | --- |
| Stu\_Id | Stu\_Name | Stu\_Age |
| 111 | Ashish | 23 |
| 123 | Saurav | 22 |
| 169 | Lester | 24 |
| 234 | Lou | 26 |

**Table: Course**

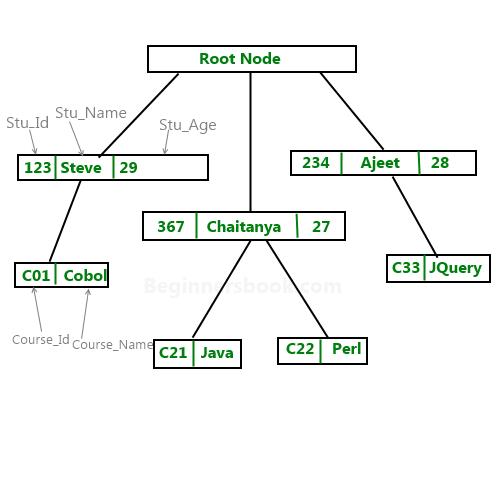
|  |  |  |
| --- | --- | --- |
| Stu\_Id | Course\_Id | Course\_Name |
| 111 | C01 | Science |
| 111 | C02 | DBMS |
| 169 | C22 | Java |
| 169 | C39 | Computer Networks |

Here Stu\_Id, Stu\_Name & Stu\_Age are attributes of table Student and Stu\_Id, Course\_Id & Course\_Name are attributes of table Course. The rows with values are the records (commonly known as tuples).

# Hierarchical model in DBMS

In **hierarchical model**, data is organized into a tree like structure with each record is having one parent record and many children. The main drawback of this model is that, it can have only one to many relationships between nodes.

**Note: Hierarchical models are rarely used now.**

**Sample Hierarchical Model Diagram**:  
Lets say we have few students and few courses and a course can be assigned to a single student only, however a student take any number of courses so this relationship becomes one to many.  


**Example of hierarchical data represented as relational tables:** The above hierarchical model can be represented as relational tables like this:

|  |  |  |
| --- | --- | --- |
| Stu\_Id | Stu\_Name | Stu\_Age |
| 123 | Steve | 29 |
| 367 | Chaitanya | 27 |
| 234 | Ajeet | 28 |

Course Table:

|  |  |  |
| --- | --- | --- |
| Course\_Id | Course\_Name | Stu\_Id |
| C01 | Cobol | 123 |
| C21 | Java | 367 |
| C22 | Perl | 367 |
| C33 | JQuery | 234 |

# Constraints in DBMS

Constraints enforce limits to the data or type of data that can be inserted/updated/deleted from a table. The whole purpose of constraints is to maintain the **data integrity**during an update/delete/insert into a table. In this tutorial we will learn several types of constraints that can be created in RDBMS.

## Types of constraints

* NOT NULL
* UNIQUE
* DEFAULT
* CHECK
* Key Constraints – PRIMARY KEY, FOREIGN KEY
* Domain constraints
* Mapping constraints

#### NOT NULL:

NOT NULL constraint makes sure that a column does not hold NULL value. When we don’t provide value for a particular column while inserting a record into a table, it takes NULL value by default. By specifying NULL constraint, we can be sure that a particular column(s) cannot have NULL values.

Example:

CREATE TABLE STUDENT(

ROLL\_NO INT **NOT NULL**,

STU\_NAME VARCHAR (35) **NOT NULL**,

STU\_AGE INT **NOT NULL**,

STU\_ADDRESS VARCHAR (235),

PRIMARY KEY (ROLL\_NO)

);

#### UNIQUE:

UNIQUE Constraint enforces a column or set of columns to have unique values. If a column has a unique constraint, it means that particular column cannot have duplicate values in a table.

CREATE TABLE STUDENT(

ROLL\_NO INT NOT NULL,

STU\_NAME VARCHAR (35) NOT NULL **UNIQUE**,

STU\_AGE INT NOT NULL,

STU\_ADDRESS VARCHAR (35) **UNIQUE**,

PRIMARY KEY (ROLL\_NO)

);

#### DEFAULT:

The DEFAULT constraint provides a default value to a column when there is no value provided while inserting a record into a table.

CREATE TABLE STUDENT(

ROLL\_NO   INT  NOT NULL,

STU\_NAME VARCHAR (35) NOT NULL,

STU\_AGE INT NOT NULL,

EXAM\_FEE INT  **DEFAULT** 10000,

STU\_ADDRESS VARCHAR (35) ,

PRIMARY KEY (ROLL\_NO)

);

#### CHECK:

This constraint is used for specifying range of values for a particular column of a table. When this constraint is being set on a column, it ensures that the specified column must have the value falling in the specified range.

CREATE TABLE STUDENT(

ROLL\_NO   INT  NOT NULL CHECK(ROLL\_NO >1000) ,

STU\_NAME VARCHAR (35)  NOT NULL,

STU\_AGE INT  NOT NULL,

EXAM\_FEE INT DEFAULT 10000,

STU\_ADDRESS VARCHAR (35) ,

PRIMARY KEY (ROLL\_NO)

);

In the above example we have set the check constraint on ROLL\_NO column of STUDENT table. Now, the ROLL\_NO field must have the value greater than 1000.

## Key constraints:

#### PRIMARY KEY:

[Primary key](https://beginnersbook.com/2015/04/primary-key-in-dbms/) uniquely identifies each record in a table. It must have unique values and cannot contain nulls. In the below example the ROLL\_NO field is marked as primary key, that means the ROLL\_NO field cannot have duplicate and null values.

CREATE TABLE STUDENT(

ROLL\_NO   INT  NOT NULL,

STU\_NAME VARCHAR (35)  NOT NULL UNIQUE,

STU\_AGE INT NOT NULL,

STU\_ADDRESS VARCHAR (35) UNIQUE,

**PRIMARY KEY** (ROLL\_NO)

);

#### FOREIGN KEY:

Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

#### Domain constraints:

Each table has certain set of columns and each column allows a same type of data, based on its data type. The column does not accept values of any other data type.  
[Domain constraints](https://beginnersbook.com/2015/04/domain-constraints-in-dbms/) are **user defined data type** and we can define them like this:

Domain Constraint = data type + Constraints (NOT NULL / UNIQUE / PRIMARY KEY / FOREIGN KEY / CHECK / DEFAULT)

# RDBMS Concepts

**RDBMS** stands for relational database management system. A relational model can be represented as a table of rows and columns. A relational database has following major components:  
1. Table  
2. Record or Tuple  
3. Field or Column name or Attribute  
4. Domain  
5. Instance  
6. Schema  
7. Keys

## 1. Table

A table is a collection of data represented in rows and columns. Each table has a name in database. For example, the following table “STUDENT” stores the information of students in database.

**Table: STUDENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student\_Id** | **Student\_Name** | **Student\_Addr** | **Student\_Age** |
| 101 | Chaitanya | Dayal Bagh, Agra | 27 |
| 102 | Ajeet | Delhi | 26 |
| 103 | Rahul | Gurgaon | 24 |
| 104 | Shubham | Chennai | 25 |

## 2. Record or Tuple

Each row of a table is known as record. It is also known as tuple. For example, the following row is a record that we have taken from the above table.

|  |  |  |  |
| --- | --- | --- | --- |
| 102 | Ajeet | Delhi | 26 |

## 3. Field or Column name or Attribute

The above table “STUDENT” has four fields (or attributes): Student\_Id, Student\_Name, Student\_Addr & Student\_Age.

## 4. Domain

A domain is a set of permitted values for an attribute in table. For example, a domain of month-of-year can accept January, February,…December as values, a domain of dates can accept all possible valid dates etc. We specify domain of attribute while creating a table.

An attribute cannot accept values that are outside of their domains. For example, In the above table “STUDENT”, the Student\_Id field has integer domain so that field cannot accept values that are not integers for example, Student\_Id cannot has values like, “First”, 10.11 etc.

## 5. Instance and Schema

## 6. Keys