WILEY EDGE

RDBMS

Contents

Relational Database Management System

- A relational database is a type of database that stores and provides access to data points that are related to one another. Relational databases are based on the relational model, an intuitive, straightforward way of representing data in tables.
- In a relational database, each row in the table is a record with a unique ID called the key. The columns of the table hold attributes of the data, and each record usually has a value for each attribute, making it easy to establish the relationships among data points.

ACID Properties

- Atomicity
- Consistency
- Isolation
- Durability

Explanation

ProductID	Name	Price	Quantity
101	Laptop	15000	100
102	Desktop	20000	150
103	Mobile	3000	200
104	Tablet	4000	250

Product Table

ProductSalesId	Produc	QuantitySold
1	101	10
2	102	15
3	103	30
4	104	35

ProductSales Table

- Atomicity
- Consistency
- Isolation
- Durability

ER Diagram

Entity Relationship Diagram (ER Diagram or ERD) is a pictorial or visual representation of classifying groups or entities of common interest and defining the relationship between these groups

Components of ER Diagram

- Entity
  - Any object that can have data stored in it.
- Relationship between the entities
  - Defines how the entities are associated or related with each other
- Attributes of entities and relationship
  - Represents the characteristics or property of an entity

Entity Key

- Attribute of an entity that helps to define an entity uniquely, within an entity set.
  - Super key
  - Primary or Candidate key
  - Foreign Key

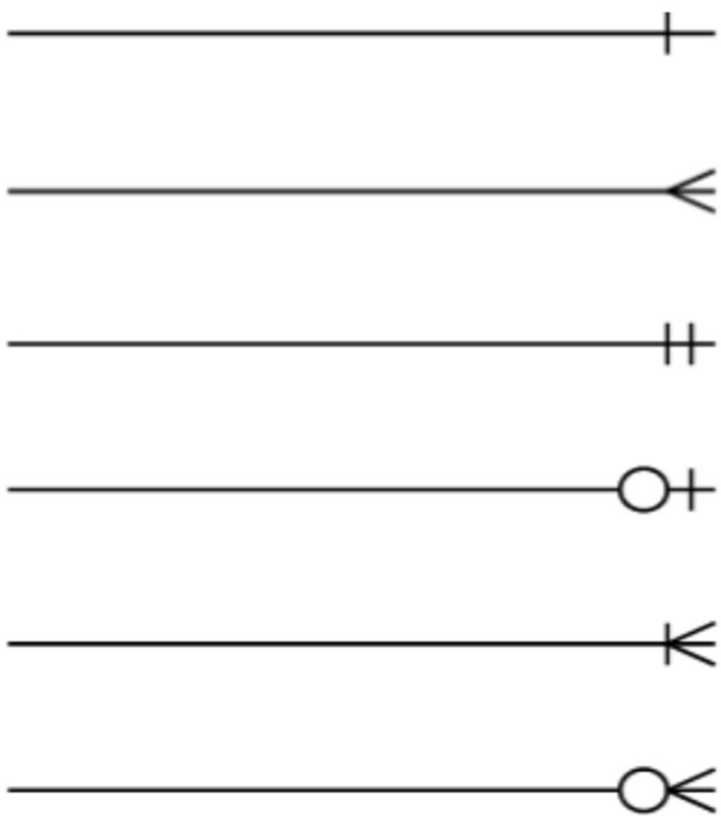
Cardinality

- One to One Relationship
  - One student can have only one Scholarship
- One to Many Relationship
  - One student can have multiple salary slips.
- Many to One Relationship
  - Many students registers to one course
- Many to Many Relationship
  - Many employees can work in Many shifts

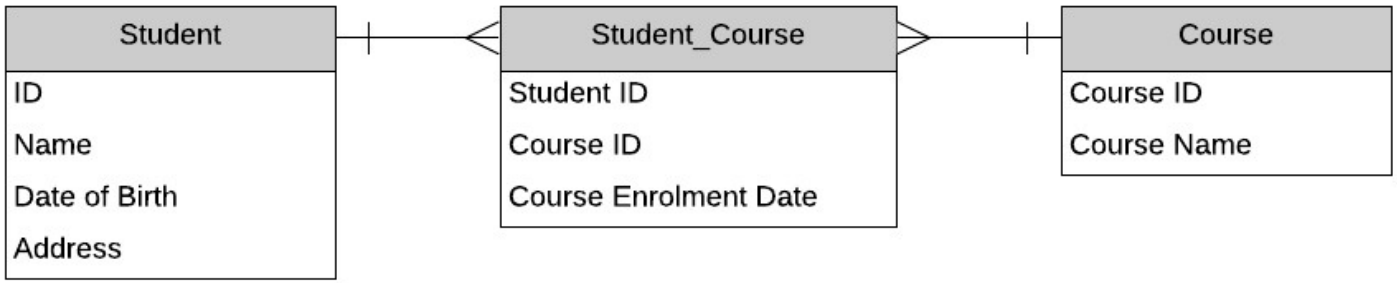
ER diagram notation types

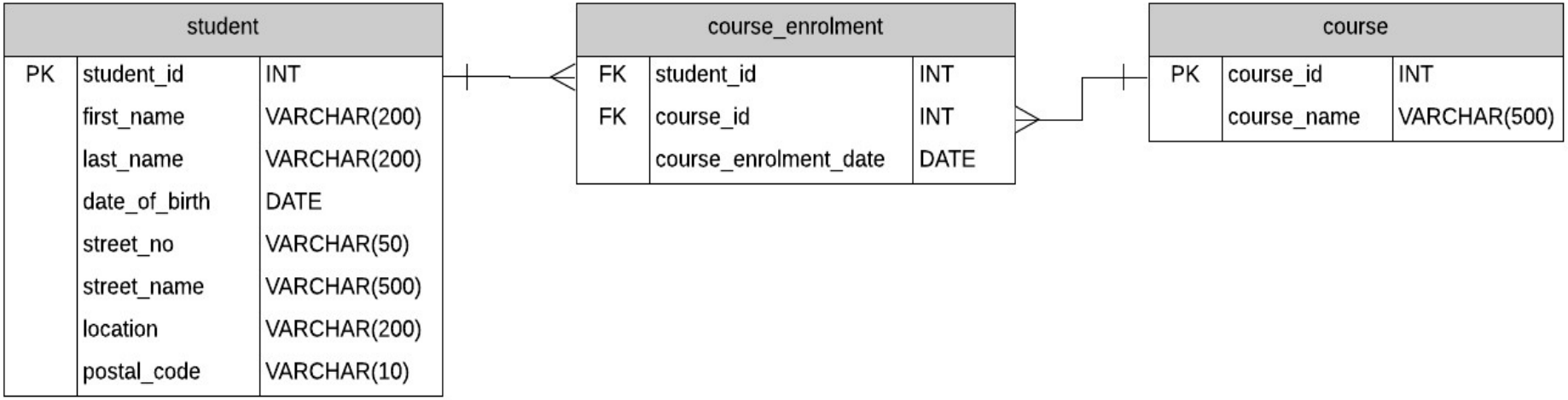
- Chen Notation Style
- Crows Foot Notation Style
- Bachman Notation Style
- IDEF1X Notation Style
- Barker Notation Style

Cardinality Notations



Logical Data Modelling vs Physical Data Modelling





Functional Dependencies

- **Functional Dependency (FD)** is a constraint that determines the relation of one attribute to another attribute in a Database Management System (DBMS).
- A functional dependency is denoted by an arrow "→". The functional dependency of X on Y is represented by X → Y.
- Types of Functional Dependencies
  - Multivalued Dependency
  - Trivial Functional Dependency
  - Non Trivial Functional Dependency
  - Transitive Depencency

Need for Normalization

- Reducing the amount of storage needed to store the data.
- Avoiding unnecessary data conflicts that may creep in because of multiple copies of the same data getting stored.

Course code	Course venue	Instructor Name	Instructor's phone number
CS101	Lecture Hall 20	Prof. George	+1 6514821924
CS152	Lecture Hall 21	Prof. Atkins	+1 6519272918
CS154	CS Auditorium	Prof. George	+1 6514821924

Insturctor's ID	Instructor's name	Instructor's number
1	Prof. George	+1 6514821924
2	Prof. Atkins	+1 6519272918

Instructor

Course code	Course venue	Instructor ID
CS101	Lecture Hall 20	1
CS152	Lecture Hall 21	2
CS154	CS Auditorium	1

Course

First Normal Form (1NF)

Each cell of the table should contain only one value.

Instructor's name	Course code
Prof. George	(CS101, CS154)
Prof. Atkins	(CS152)

Instructor's name	Course code
Prof. George	CS101
Prof. George	CS154
Prof. Atkins	CS152

Second Normal Form (2NF)

Table should be in the first normal form

The primary key of the table must be exactly one column

Course code	Course venue	Instructor Name	Instructor's phone number
CS101	Lecture Hall 20	Prof. George	+1 6514821924
CS152	Lecture Hall 21	Prof. Atkins	+1 6519272918
CS154	CS Auditorium	Prof. George	+1 6514821924

Student name	Course code
Rahul	CS152
Rajat	CS101
Rahul	CS154
Raman	CS101

Student name	Enrolment number
Rahul	1
Rajat	2
Raman	3

Course code	Enrolment number
CS101	2
CS101	3
CS152	1
CS154	1

Third Normal Form (3NF)

Used when there is functional dependency in table columns
Table should be in 2NF

There should not be functional dependency between columns

Course code	Course venue	Instructor's name	Department
MA214	Lecture Hall 18	Prof. George	CS Department
ME112	Auditorium building	Prof. John	Electronics Department

Course code	Course venue	Instructor's ID
MA214	Lecture Hall 18	1
ME112	Auditorium building,	2

Instructor's ID	Instructor's Name	Department
1	Prof. Ronald	Mathematics Department
2	Prof. John	Electronics Department

Order by

Syntax

```
SELECT
  select_list
FROM
  table_name
ORDER BY
  column1 [ASC|DESC],
  column2 [ASC|DESC],
  ...;
```

Things you should know…

- show databases
- Use <database>
- Show tables
- The dual table
  - Select database() from dual;
  - select upper('wiley') from dual
  - Sysdate()
  - Current_date()

Types of Queries

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Transaction Control Language (TCL)
- Data Query Language (DQL)

Creating table

Syntax

```
CREATE TABLE [IF NOT EXISTS] table_name(
   column_1_definition,
   column_2_definition,
   ...,
   table_constraints
);
```

Inserting values

```
INSERT INTO table(c1,c2,...)
VALUES
   (v11,v12,...),
   (v21,v22,...),
    ...
   (vnn,vn2,...);
```

Where clause

Syntax

```
SELECT
    select_list
FROM
    table_name
WHERE
    search_condition;
```

Distint Clause

Syntax

```
SELECT DISTINCT
    select_list
FROM
    table_name
WHERE
    search_condition
ORDER BY
    sort_expression;
```

MySQL Constraints

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

General Syntax of Table Creation

```
CREATE TABLE [table name] (
[column name] [data type]([size]) [column constraint]…. [
table constraint] ([[column name]……])……
);
```

Using Null constraint

```
CREATE TABLE IF NOT EXISTS Author (
aut_id varchar(8) NOT NULL,
aut_name varchar(50) NOT NULL );
```

Using check constraint

```
CREATE TABLE IF NOT EXISTS Book(
  book_id varchar(15) NOT NULL UNIQUE,
  book_name varchar(50) , isbn_no varchar(15) NOT NULL UNIQUE ,
  cate_id varchar(8) ,
  aut_id varchar(8) , pub_id varchar(8) , dt_of_pub date , pub_lang varchar(15) ,
  no_page decimal(5,0) CHECK(no_page>0) ,
  book_price decimal(8,2));
```

Using IN

```
CREATE TABLE IF NOT EXISTS Author(aut_id varchar(8) NOT NULL ,
  aut_name varchar(50) NOT NULL,
  country varchar(25) NOT NULL CHECK (country IN ('USA','UK','India')),
  home_city varchar(25) NOT NULL);
```

Using unique

```
CREATE TABLE IF NOT EXISTS Author(aut_id varchar(8) NOT NULL ,
  aut_name varchar(50) NOT NULL,
  country varchar(25) NOT NULL,
  home_city varchar(25) NOT NULL, UNIQUE (aut_id));
```

Default Constraint

```
CREATE TABLE IF NOT EXISTS publisher (
  pub_id varchar(8) NOT NULL UNIQUE DEFAULT ‘’ ,
  pub_name varchar(50) NOT NULL DEFAULT ‘’ ,
  pub_city varchar(25) NOT NULL DEFAULT ‘’ ,
  country varchar(25) NOT NULL DEFAULT 'India’
);
```

Using Auto Increment

```
CREATE TABLE IF NOT EXISTS Author(
  id int NOT NULL AUTO_INCREMENT,
  aut_id varchar(8),
  aut_name varchar(50),
  country varchar(25),
  home_city varchar(25) NOT NULL,
 )
```

Using Primary Key

```
CREATE TABLE IF NOT EXISTS Author(
  aut_id varchar(8) NOT NULL ,
  aut_name varchar(50) NOT NULL,
  country varchar(25) NOT NULL,
  home_city varchar(25) NOT NULL,
  PRIMARY KEY (aut_id)
);
```

Using Primary key on multiple Columns

```
CREATE TABLE IF NOT EXISTS
  newauthor(aut_id varchar(8) NOT NULL ,
  aut_name varchar(50) NOT NULL,
  country varchar(25) NOT NULL,
  home_city varchar(25) NOT NULL,
  PRIMARY KEY (aut_id, home_city));
```

Using Foreign Key

Syntax

```
FOREIGN KEY [column list] REFERENCES [primary key table] ([column list]);
```

Name	Description
column list	A list of the columns on which FOREIGN KEY is to be set.
REFERENCES	Keyword.
primary key table	Table name which contains the PRIMARY KEY.
column list	A list of the columns on which PRIMARY KEY is set in the primary key table.

```
CREATE TABLE IF NOT EXISTS book_mast (
  book_id varchar(15) NOT NULL PRIMARY KEY,
  no_page decimal(5,0) ,
  book_price decimal(8,2) ,
  FOREIGN KEY (aut_id) REFERENCES Author(aut_id));
```

Using Cascade and Restrict

```
CREATE TABLE IF NOT EXISTS purchase (
  invoice_no varchar(12) NOT NULL UNIQUE PRIMARY KEY,
  invoice_dt date ,
  ord_no varchar(25) , ord_date date , receive_dt date ,
  book_id varchar(8) , book_name varchar(50) ,
  FOREIGN KEY(ord_no,book_id) REFERENCES  neworder(ord_no,book_id)
  ON UPDATE CASCADE ON DELETE RESTRICT,
  FOREIGN KEY(cate_id) REFERENCES category(cate_id));
```