NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

1.2 Introduction to Database

Akanksha Tiwari

11 Jan 2025

Module Overview

1.1 Introduction to Data Science
1.2 Introduction to Database
1.3 SQL Basic - DDL
1.4 SQL Basic - DML
1.5 SQL Advanced
1.6 Introduction to Numpy
1.7 Introduction to Pandas
1.8 Exploratory Data Analysis (EDA) Basic
1.9 EDA Advanced
1.10 Data Visualisation

Agenda

- Introduction to database
- Data modeling

Database Management System (DBMS)

Recap: *A database is a collection of data organized in a structured way. It can be accessed or stored in a computer system and managed through a Database Management System (DBMS).*

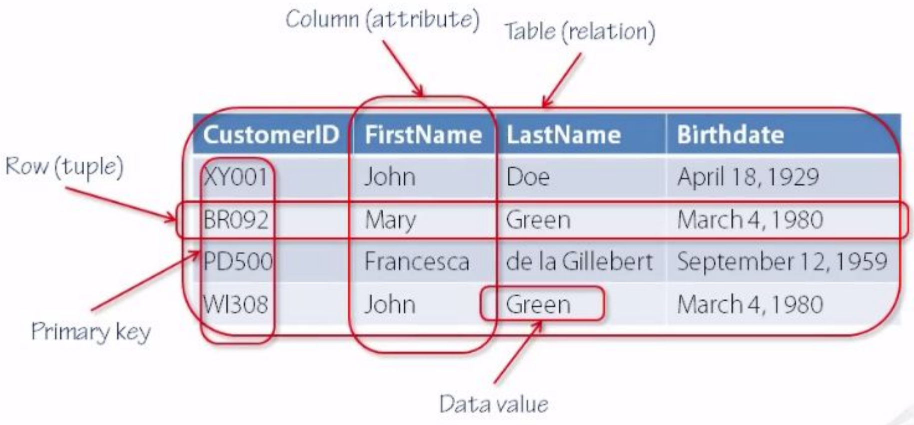
Databases are used to store and manage data for various applications such as websites, mobile apps, and enterprise systems.

Relational Database Management System (RDBMS)

- Historical Context:
  - Described in 1969 by English computer scientist Edgar F. Codd.
- ***All data is represented in terms of rows, grouped into tables (or relations).***
- Databases that implement the relational model are often referred to as relational databases.
- A relational database management system (RDBMS) is a program that allows you to create, update, and administer a relational database.
- Popular databases- *MySQL, PostgreSQL, SQL Server, SQLite, Oracle DB*

Relational Database I

- Data in a relational database is organized into *tables or relations*.
- Tables can have hundreds to millions of *rows* of data. These rows are often called *records or tuples*.
- Tables can also have many columns of data. Columns are labeled with a descriptive name (say, age for example) and have a specific *data type*.

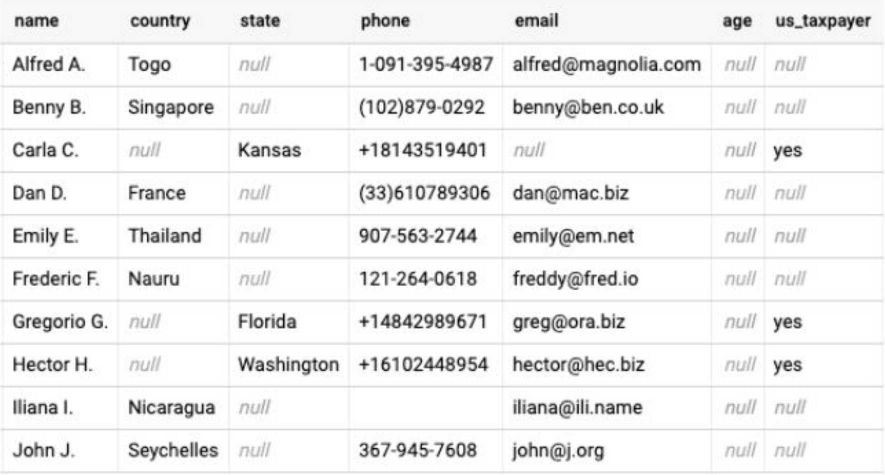


CustomerID	FirstName	LastName	Birthdate
XY001	John	Doe	April 18, 1929
BR092	Mary	Green	March 4, 1980
PD500	Francesca	de la Gillebert	September 12, 1959
WI308	John	Green	March 4, 1980

Column (attribute)
Table (relation)
Row (tuple)
Primary key
Data value

Relational Database II

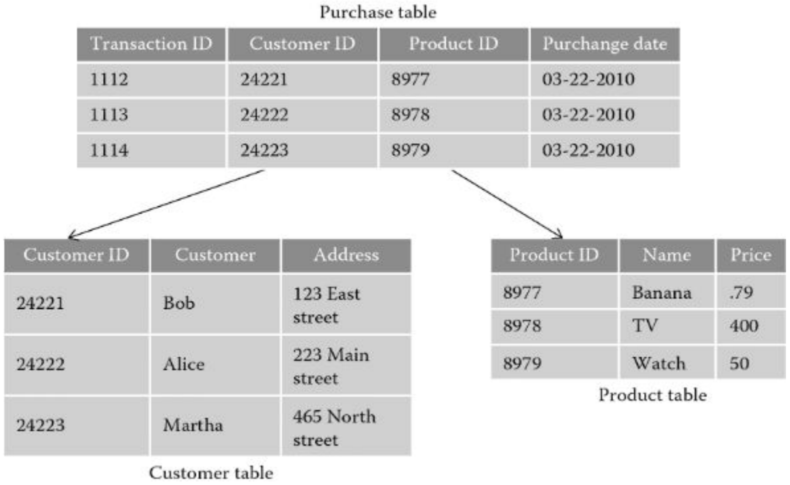
- Each column has a specific data type such as varchar (text), integer, float etc.
  - Regardless of the data type, empty value can be present in the column.
  - Empty value is represented as NULL



name	country	state	phone	email	age	us_taxpayer
Alfred A.	Togo	*null*	1-091-395-4987	alfred@magnolia.com	*null*	*null*
Benny B.	Singapore	*null*	(102)879-0292	benny@ben.co.uk	*null*	*null*
Carla C.	*null*	Kansas	+18143519401	*null*	*null*	yes
Dan D.	France	*null*	(33)610789306	dan@mac.biz	*null*	*null*
Emily E.	Thailand	*null*	907-563-2744	emily@em.net	*null*	*null*
Frederic F.	Nauru	*null*	121-264-0618	freddy@fred.io	*null*	*null*
Gregorio G.	*null*	Florida	+14842989671	greg@ora.biz	*null*	yes
Hector H.	*null*	Washington	+16102448954	hector@hec.biz	*null*	yes
Iliana I.	Nicaragua	*null*		iliana@ili.name	*null*	*null*
John J.	Seychelles	*null*	367-945-7608	john@j.org	*null*	*null*

Relational Database III

- Primary key is a column or a set of columns that uniquely identifies each row in the table.
  - A table can have only one primary key.
  - Primary key must contain unique values and cannot contain NULL values.
- Foreign key is a column or a set of columns in a table that provides a link to another table. It establishes link between tables because it references the primary key of another table.



Purchase table

Transaction ID	Customer ID	Product ID	Purchange date
1112	24221	8977	03-22-2010
1113	24222	8978	03-22-2010
1114	24223	8979	03-22-2010

Customer ID	Customer	Address
24221	Bob	123 East street
24222	Alice	223 Main street
24223	Martha	465 North street

Customer table

Product ID	Name	Price
8977	Banana	.79
8978	TV	400
8979	Watch	50

Product table

Data Modeling

What is a data model?

A **data model** is a conceptual framework that defines how data is structured, organized, and managed in a database or system. It provides a blueprint for designing and implementing a database, helping to represent real-world entities, their attributes, and the relationships between them.

Data Modeling

Why is it important ?

- Business Alignment: Critical step to make data useful for the business by ensuring it reflects your organization's processes, definitions, workflows, and logic.

- Decision Support: A good data model correlates with impactful business decisions by providing a coherent structure for analysis and reporting.

Example: In an e-commerce database, the data model would define how customer information, orders, products, and inventory are structured and interconnected to support business operations and analysis.

Conceptual, Logical, and Physical Data Models

When modeling data, the idea is to move from abstract modeling concepts to concrete implementation.

- **Conceptual**- Contains business logic and rules and describes the system's data, such as *schemas, tables, and fields (names and types)*. Often visualized in an *entity-relationship diagram (ERD)*.

Conceptual, Logical, and Physical Data Models

When modeling data, the idea is to move from abstract modeling concepts to concrete implementation.

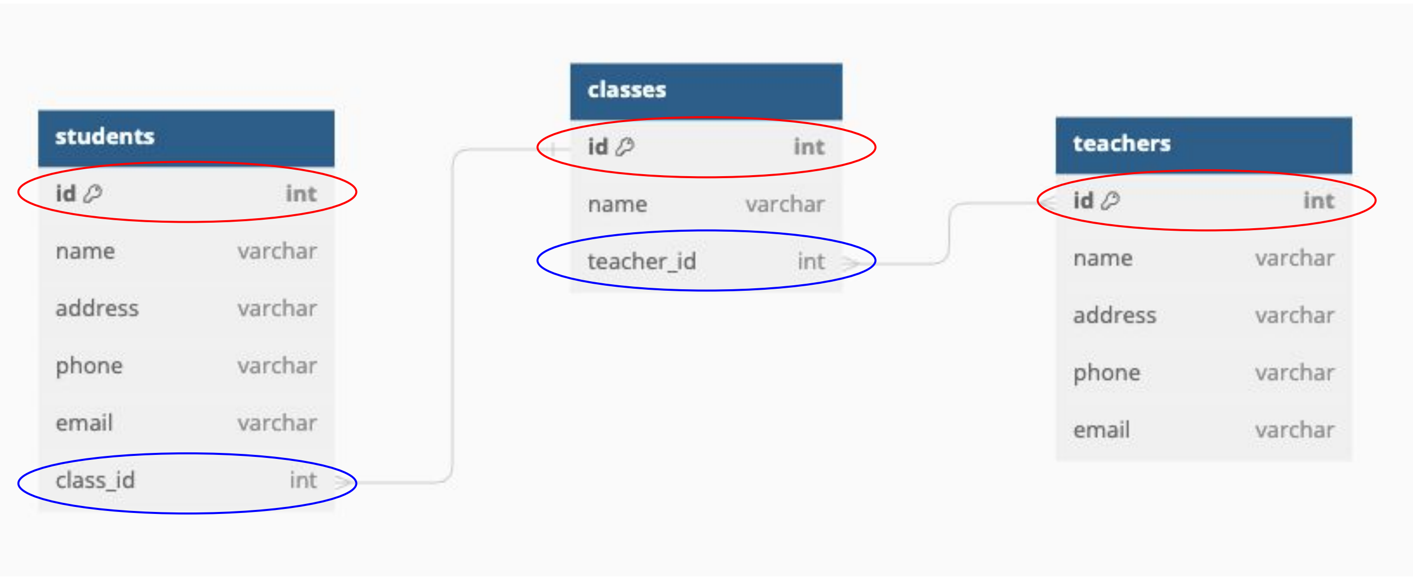
- **Conceptual**- Contains business logic and rules and describes the system's data, such as *schemas, tables, and fields (names and types)*. Often visualized in an *entity-relationship diagram (ERD)*.
- **Logical**- Details how the conceptual model will be implemented in practice by adding significantly more detail.

Conceptual, Logical, and Physical Data Models

When modeling data, the idea is to move from abstract modeling concepts to concrete implementation.

- **Conceptual**- Contains business logic and rules and describes the system’s data, such as *schemas, tables, and fields (names and types)*. Often visualized in an *entity-relationship diagram (ERD)*.
- **Logical**- Details how the conceptual model will be implemented in practice by adding significantly more detail.
- **Physical**- Defines how the logical model will be implemented in a database system. We would add specific *databases, schemas, and tables* to our logical model, including configuration details.

Entity-relationship diagram (ERD)



Primary Key

Foreign Key

students	
id 🔑	int
name	varchar
address	varchar
phone	varchar
email	varchar
class_id	int

classes	
id 🔑	int
name	varchar
teacher_id	int

teachers	
id 🔑	int
name	varchar
address	varchar
phone	varchar
email	varchar

Scenario 2

```
Table students {
  id int [pk]
  name varchar
  address varchar
  phone varchar
  email varchar
  class_id int
}

Table classes {
  id int [pk]
  name varchar
  teacher_id int
}

Table teachers {
  id int [pk]
  name varchar
  address varchar
  phone varchar
  email varchar
}
Ref: students.class_id > classes.id // A student belongs to one class
Ref: classes.teacher_id <> teachers.id // A class is taught by one teacher
```

Scenario 3

```
table customer {
  customer_id int [pk]
  name varchar
  email varchar
  phone_number varchar
  shipping_address text
}
table order {
  order_id int [pk]
  customer_id int // Foreign key to customer
  movie_id int  // Foreign key to movie
  order_date datetime
  shipping_address text
  total_price decimal
}
table movie {
  movie_id int [pk]
  title varchar
  genre varchar
  release_date date
  director varchar
  price decimal
}
ref: order.customer_id > customer.customer_id
ref: order.movie_id <> movie.movie_id
```