DBMS lecture notes

Database management systems

These are the notes for database management systems course. The professor says it'll be an easy course, let's see about that. I am using Obsidian and this is an amazing markdown editor! It has a lot of community plugins. Anyways, study now... xD

Here is a somewhat detailed overview.

1. DBMS/Lecture 1 : Introduction to the course and grading.
2. DBMS/Lecture 2 : Something more here
3. DBMS/Lecture 3 : IDK what's going on

Lectio Prima

There are very little things we did in the first lecture. One can say, we did "nothing". Here is a summary of the classroom policies.

Broad objectives

- Learn database principles
- Learn how to design databases
- Learn languages for developing database applications
  - SQL
  - Java/Any other language interfaces
- Learn some aspects of DB system internals
- About storage, indexing, transaction management, query execution, etc

Text and lecture material

- Database System Concepts, Silberschatz, Korth and Sudarshan, McGraw Hill 7^{th} Ed. Here are the Book slides here
- Fundamentals of Database Systems, Elmasri and Navathe

- Lecture slides will be provided in advanced

Evaluation Scheme

1. Midterm: 25%
2. Endterm: 35%
3. Tutorial Attendance: 2%
4. Quizzes in tutorial class: 8%
5. Project: 30%
   (a) Mid-project: 10% (During week 7, 28 Feb to 4 March)
   (b) Final-project: 15% (During week 15, 26 April to 29 April)
   (c) Project document/code submission (Week 15)

Task for today?

- Download mysql and get familiarized. Video

Remaining content is just evaluation jargon which might be *understandable* later.

Lectio Secunda

Everything is from Silberschatz

Purpose of DBS: To provide users with abstract view of data

- Data models: A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.
- Data abstraction: Hide complexity with layers

Data Models

- Entity-relationship data model (mainly for database-design)
- Relational model
- Object-based data models (Object oriented and object relational)
- Semi-structured data model (XML)
- Network model
- Heirarchical model

View of Data (Levels of Abstraction)

- View level: Can hide info. for security.
- Logical level: Describes stores data and relationship (Name: string, CustID: Integer,...)
- Physical level: Describes how a record is stored. (e.g. customer)

- Logical data independence: ability to modify logical scheme without rewriting application program
- Physical data independence: ability to modify physical schema without changing logical schema

Data Definition Language (DDL)

- Example:

```
create table customer (
                    ID      char(5),
                    name    varchar(20),
                    City    varchar(20),
                    Balance numeric(8,2));
Copy
```

- DDL compiler generates a set of table templates stored in a data dictionary
- data dictionary contains metadata (i.e. data about data)
  - Database schema
  - Integrity contraints (ID to identify)
  - Authorization: Who is allowed to access

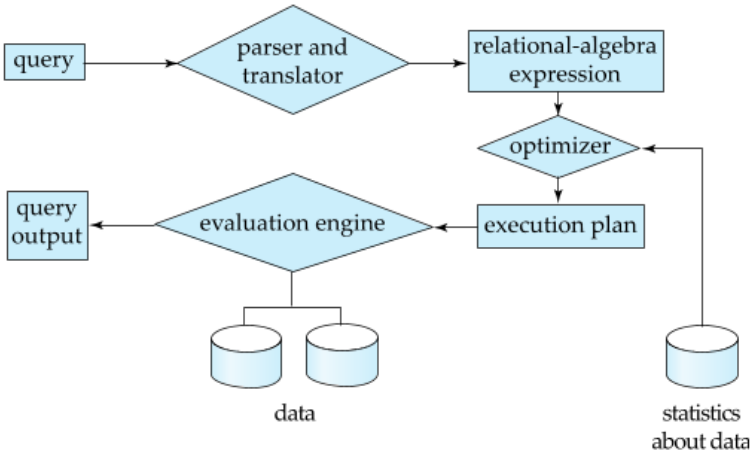
Data Manipulation Language (DML)

- Procedural DML : Require user to specify what data are needed and how to get those data. (Relational algebra?)
- Declarative/non-proc DML : Require a user to specify what data are needed without specifying how to get those data. (Easy)
  - SQL: one of more tables as input and single table as output
  - Eg: find all customers in 'Agra' whose balance $\geq$ 1Lakh.

```
select name
from customer
where city='Agra' and Balance>100000;```
- To be able to compute complex functions SQL is embedded in some higher-level language
- Application programs generally access databases through one of
    - Language extn. to allow embedded sql
    - API (e.g. ODBC/JDBC??) which allow SQL queries to be sent to database
Copy
```

Query processing

1. Parsing and translation
2. Optimization
3. Evaluation

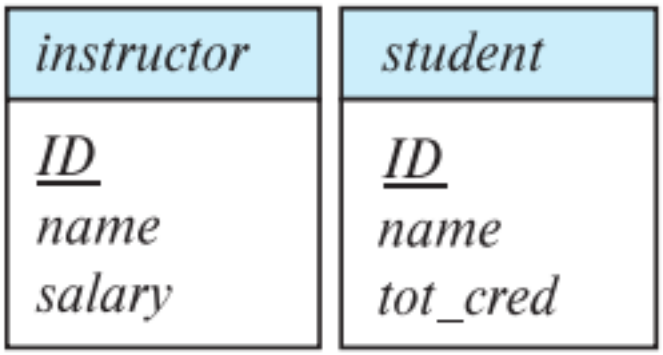


Weird lecture.

Lecture 3

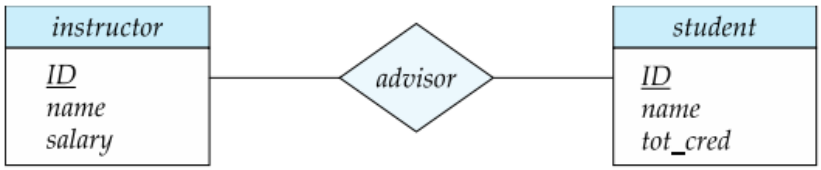
Entitties

- An entity is an object that exists and is distinguishable from other objects. Eg. specific person, company, event
- An entity set is a set of entities of the same type that share the same properties. Eg. group of people, companies, trees
  Don't come at me saying that a company is already a group of people. This was the given example lmao.
- An entity is represented by a set of attributes
  - instructor=(ID, name, salary)
  - course=(course_id, title, credits)
- A subset of attributes form a primary key of the entity set, i.e. unique identier (like Aadhar/SSN/JEE main rank+year)
- Entities in an ER diagram
  Yeah, I too regret taking this course

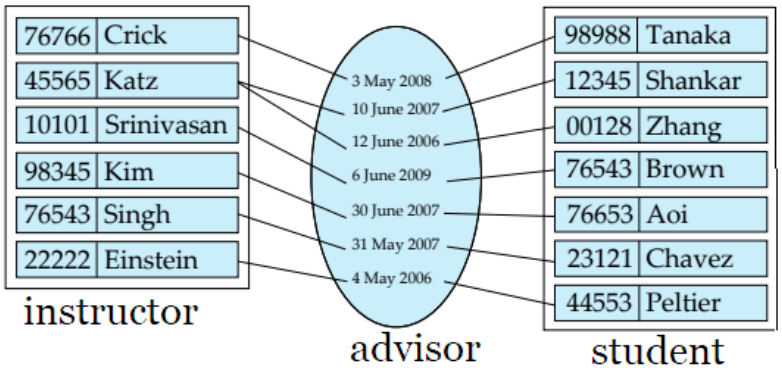


Relationship Sets

- A relationship is an association among several entities.
  E.g. Jiu Com'bak is the mentor of Mishinnhya. Here, mentor is a relationship set. It could have been, Motthya is a discord mod of n people.
- A relationship set is a mathematical relation among $n \geq 2$ entities



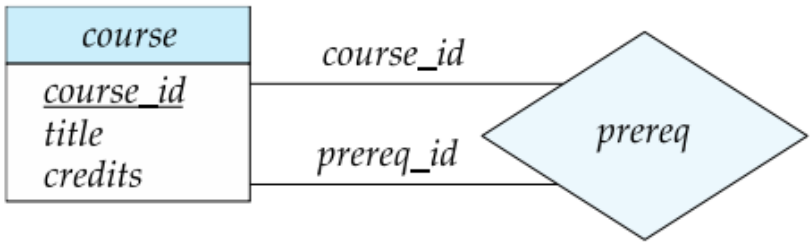
  - Now this just seems to be a label, you may add an attribute or two to it.



    For the real diagram just extend the advisor with a dashed line and write *date* in a box
- Polyamory allowed (i.e. project guide guides the prof, the student and projects)

Roles

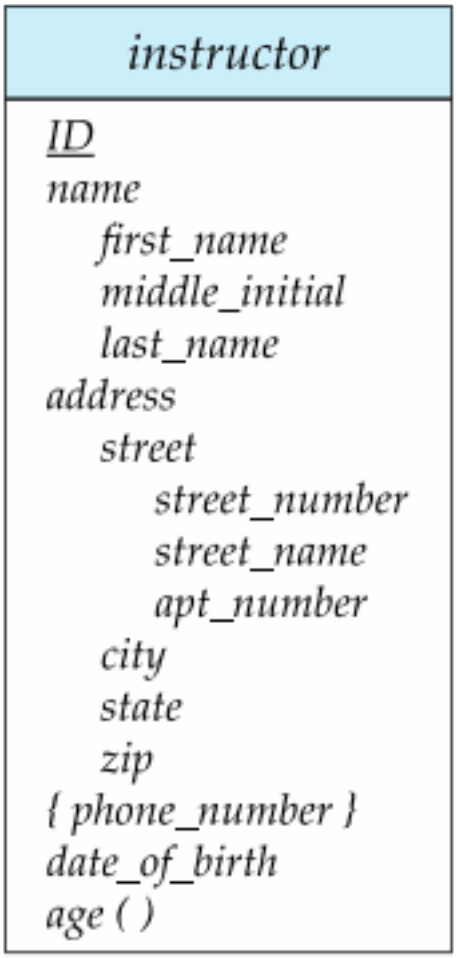
- Entity sets of a relationship need not be distinct



Here *course_id and prereq_id* are the roles. Consider discord roles (for real users), we all are human (entity set) but have different priviledge levels.

Complex Attributes

- Simple and composite (address)
- Single valued, multi-valued (phone number)
- Derived (age from D.O.B.)



What stupid UML diagram thing is this.

Cardinality Contraints

An instructor is associated with several (0 included) students via advisor
A student is associated with atmost one instructor via advisor.

Total and partial participation

The line from advisor to student when doubled (like =) becomes : An instruct associated with (1 or more) students.
Partial, as the name suggests.

– Instead of increasing lines, one can write `0..*` or `1..1` just like in UML only that here it is MATLAB notation and we have to write it on both sides of the line.

- Instructor can advise 0 or more students. A student must have 1 advisor; cannot have multiple advisors

....
Writing DBMS notes seems stupid. Decision to be taken in next lecture.

DBMS/Lecture 4