Lesson 2 - Fundamentals of Databases

2018-08-23

Models → Means of communication, users must have certain knowledge in common

↳ Only emphasize certain aspects, is described in some language, can be erroneus, may have features that do not exist in reality

When to use / not use DBMS

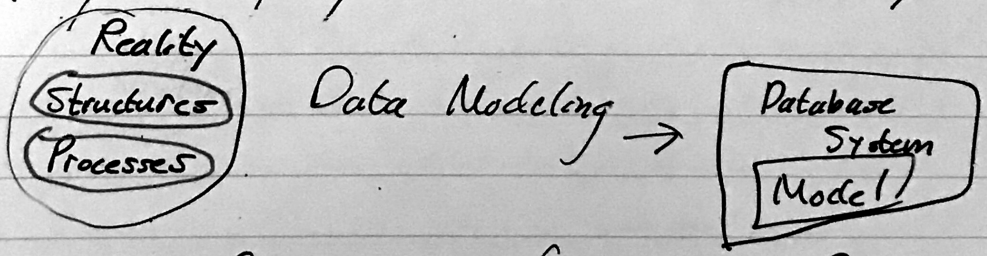
Use: Data intenoive apps, persistent storage, centralized control, control redundancy, control cosistency and integrity, multiple user support, sharing of data, data documentation data independence, control of access / security

Don't use: Initial investment in hardware, software, and training is too high, generality is not needed, data and applications are simple and stable, mult-user access is not needed

* Major Topics of course: Data modeling, process modeling, database efficiency

Data Modeling:

Model represents perception of structures of reality:



(What's the difference EERM vs. RM)

? Data Modeling: fix perception of structures of reality and represent them. Two languages: extended entity relationship model, and relational model

Summary: Models are useful for describing real-life concepts. Database management systems are ways to actualize those models, but they sometimes aren't needed. Major topics are data modeling, process modeling, and database efficiency.

2018-08-23

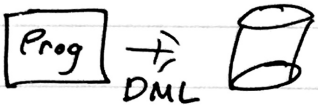
Lesson 2 – Fundamentals of Databases

DML: Data Manipulation Language (SQL)

Process Modeling: Fixing and representing perception of processes and reality.

→ Processes may be represented: (1) embedded in program code, and executed ad-hoc

Constructed query



ad-hoc query

Later Topics

Data Models, Database Architecture, Database Management SA

Data models: data structures, constraints, operations, keys and identifiers, integrity and consistency, null values, surrogates

Database architecture: ANSI/SPARC 3-level DB arch., and database metadata

Same entity as Dotnet Entity Framework

Data Models: data model not the same as a model of data.

- Entity-relationship model: we'll use for fixing perceptions
- Relational model: we'll use to implement ERM
- Hierarchical model: used in first database, XML based on this

Relational Model - Data structures:

Column name → Email, BirthDate, Hometown, Salary

Table name → RegularUser

columns

rows → user1 | 1985-01-11 | Portland | 10,000
user2 | 1986-03-14 | Austin | 13,000

} schema = stable

} state = dynamic

Summary: two types of queries (1) constructed from a program, (2) ad-hoc. SQL is a type of data manipulation language. There are three types of data models, but only two (ERM and RelM) are useful. Relational tables have columns, rows, and a name. The table schema is stable, but the state is dynamic and changing

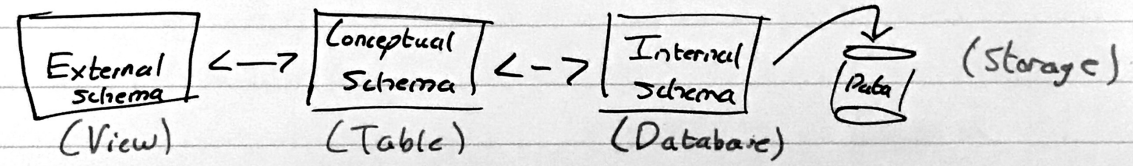
2018-08-23

Lesson 2 - Fundamentals of Databases

ANSI/SPARC 3-Level DB => Separating Concerns: A database is divided into schema and data
→ Schema describes the intension (type)
→ Data describes the extension (data)

* → The Ansi/Sparc 3-level DB architecture separates out aspects of how data is physically organized into an internal schema. This means you can change the way data is stored without affecting applications that run against the schema.

* Different applications have different needs. Therefore, applications reference external schema, which contains only the data the application needs.



* External schema: use of data
Conceptual schema: meaning of data
Internal schema: storage of data

Conceptual Schema: Describe all conceptually relevant, general, time-invariant structural aspects of reality
→ Excludes aspects of data representation and physical organization, and access

External Schema: Describes parts of the information in the conceptual schema in a form convienient to a particular user's view, and is derived from the conceptual schema

---

Summary: 3-level separates out concerns into external, conceptual, and internal schema. This allows for physical data independence (changing internal schema without changing conceptual)

2018-08-23

Lesson 2 - Fundamentals of Databases

Relational Model - Constraints: Express rules that cannot be expressed by the data structures alone:
Examples: (1) Emails must be unique, (2) Emails not allowed to be NULL, (3) Birthdate must be after 1900-01-01, and (4) Hometown must be cities in the US
↑ None of these constraints are expressed in the data itself
Data Model - Operations: Support change and retrieval of data: INSERT into `RegularUser` (. . . .);
Select from `RegularUser` Where ...;
Keys and Identifiers: Keys are uniqueness constraints

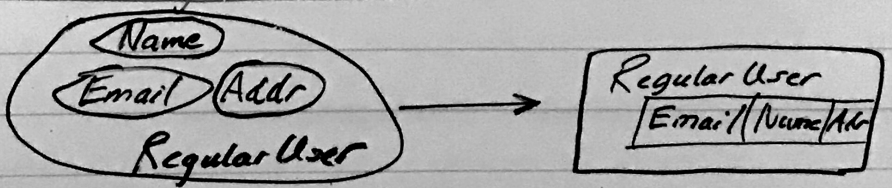
Integrity and Consistency: Integrity → does the database reflect reality well, and Consistency → Is the database without internal conflicts?

Example consistency problem: User's current city in "RegularUser" table is different from their address in "User" table
Example integrity problem: User's name in database does not reflect reality (like gender)

* Watch out for "catch-all" form

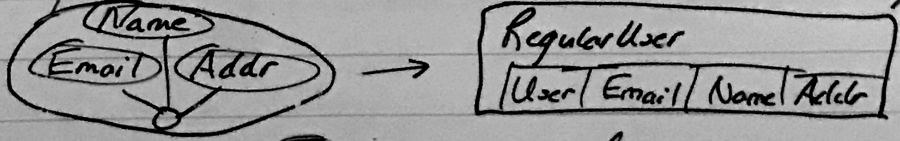
NULL Values: Applicable when one column is optional, not known for sure, or doesn't apply to that case

Surrogates - Things and Names:



Problem: A thing is what we know about it

Name-based representation: You are what is known about you



The thing exists, despite what we

Surrogate representation: System-generated, unique internal identifiers