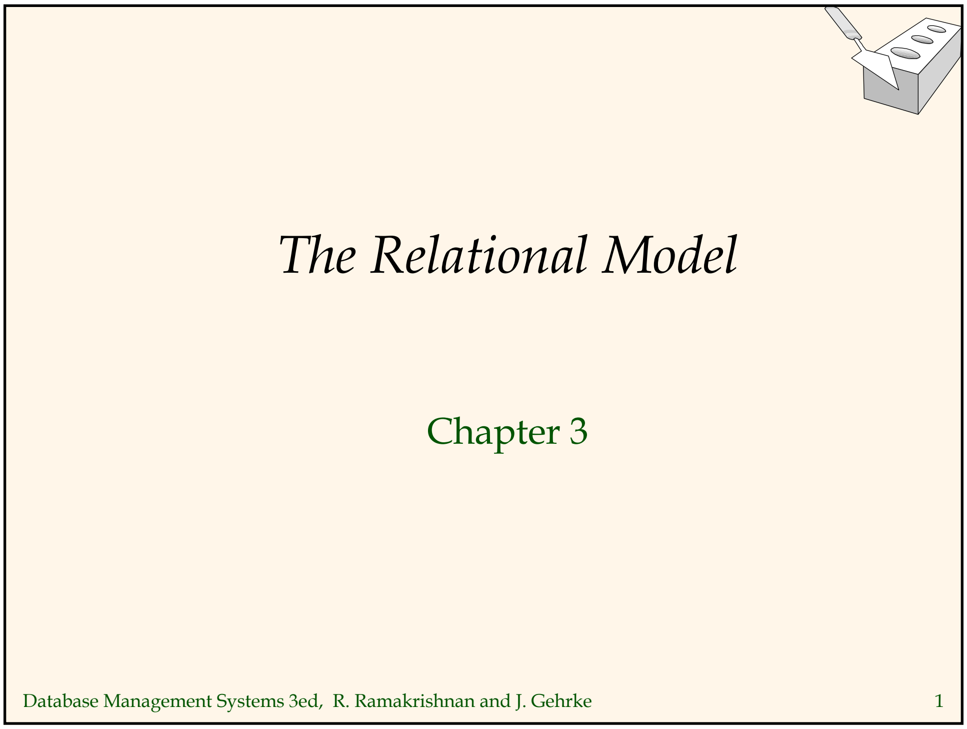
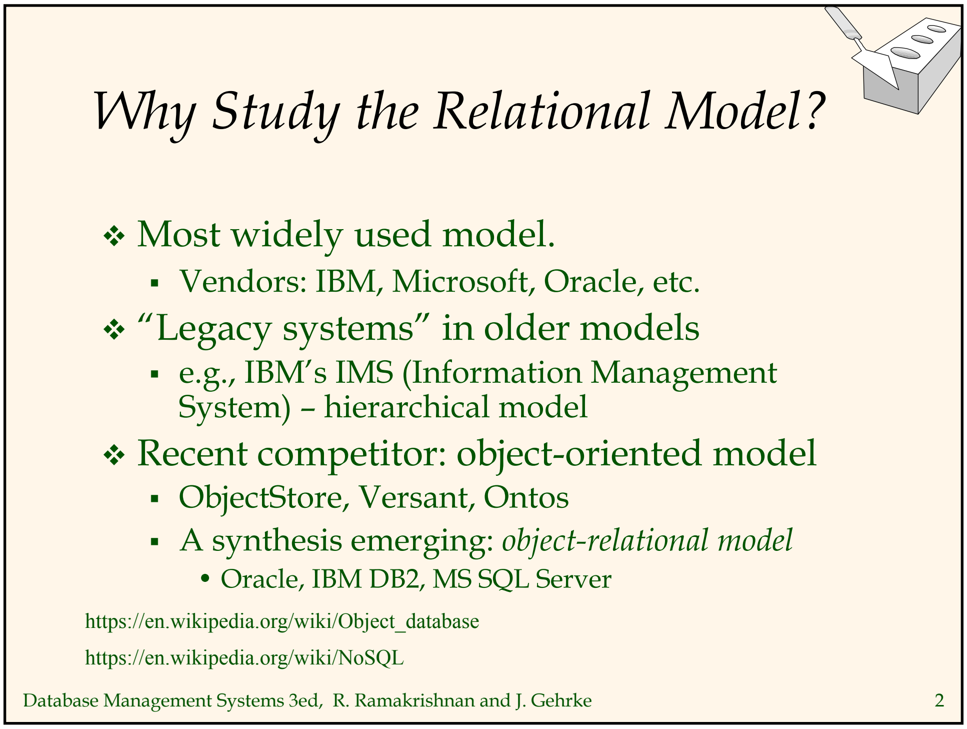


The Relational Model

Chapter 3

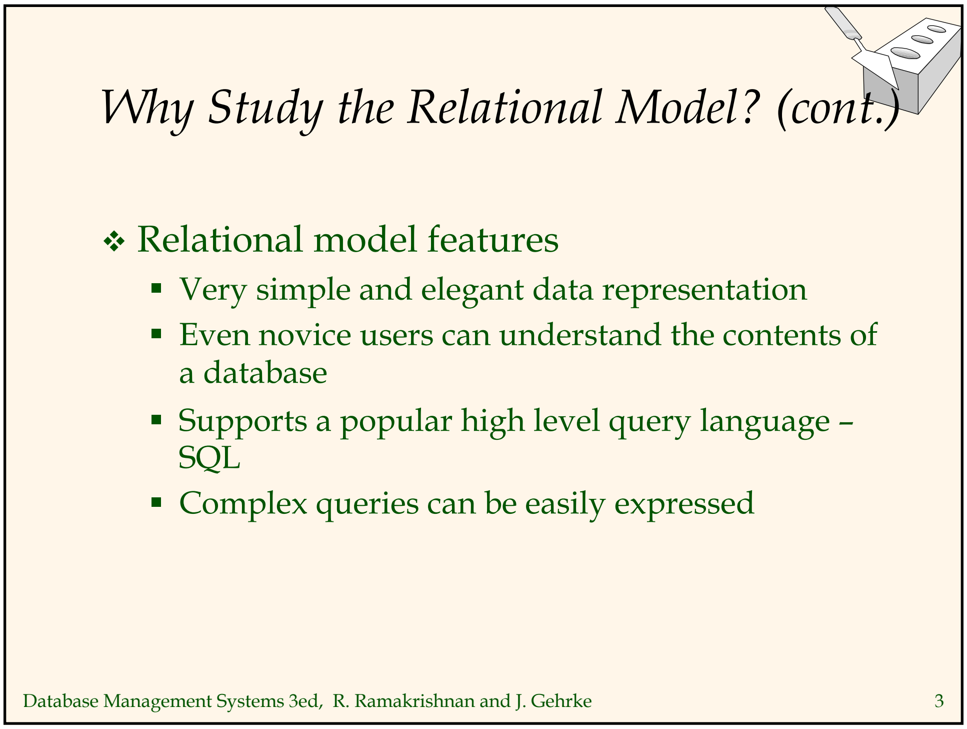


Why Study the Relational Model?

- Most widely used model.
  - Vendors: IBM, Microsoft, Oracle, etc.
- "Legacy systems" in older models
  - e.g., IBM's IMS (Information Management System) – hierarchical model
- Recent competitor: object-oriented model
  - ObjectStore, Versant, Ontos
  - A synthesis emerging: *object-relational model*
    - Oracle, IBM DB2, MS SQL Server

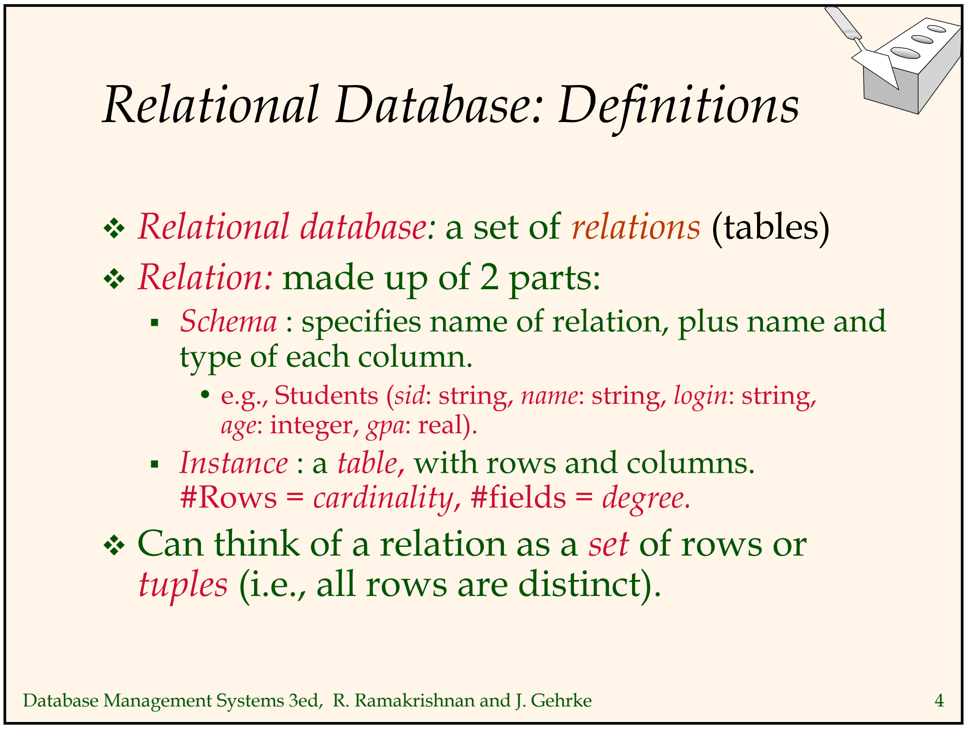
https://en.wikipedia.org/wiki/Object_database

https://en.wikipedia.org/wiki/NoSQL



Why Study the Relational Model? (cont.)

- Relational model features
  - Very simple and elegant data representation
  - Even novice users can understand the contents of a database
  - Supports a popular high level query language – SQL
  - Complex queries can be easily expressed



Relational Database: Definitions

- *Relational database:* a set of *relations* (tables)
- *Relation:* made up of 2 parts:
  - *Schema* : specifies name of relation, plus name and type of each column.
    - e.g., Students (*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real).
  - *Instance* : a *table*, with rows and columns. #Rows = *cardinality*, #fields = *degree.*
- Can think of a relation as a *set* of rows or *tuples* (i.e., all rows are distinct).

Example Instance of Students Relation

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8

- Cardinality = 3, degree = 5, all rows distinct
- Do all columns in a relation instance have to be distinct?

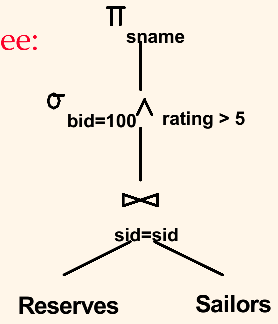
Relational Query Languages

- A major strength of the relational model: supports simple, powerful *querying* of data.
- Queries can be written intuitively, and the DBMS is responsible for efficient evaluation.
  - The key: precise semantics for relational queries.
  - Allows the optimizer to extensively re-order operations, and still ensure that the answer does not change (Chapter 12).

Overview of Query Evaluation

```
SELECT  S.sname
FROM  Reserves R, Sailors S
WHERE  R.sid=S.sid AND
   R.bid=100 AND S.rating>5
```

RA Tree:



The SQL Query Language

- Developed by IBM (for the pioneering system - System R) in the 1970s
- Need for a standard since it is used by many vendors
- Standards:
  - SQL-86
  - SQL-89 (minor revision)
  - SQL-92 (major revision)
  - SQL-1999 (major extensions)
  - SQL-2003 (minor revision)
  - SQL-2008 (minor revision)
  - SQL-2011 (minor revision)
  - SQL-2016 (latest release)

Creating Relations in SQL

- Creates the Students relation. Observe that the type (domain) of each field is specified, and enforced by the DBMS whenever tuples are added or modified.
- As another example, the Enrolled table holds information about courses that students take.

```
CREATE TABLE Students
          (sid: CHAR(20),
           name: CHAR(20),
           login: CHAR(10),
           age: INTEGER,
           gpa: REAL)
```

```
CREATE TABLE Enrolled
          (sid: CHAR(20),
           cid: CHAR(20),
           grade: CHAR(2))
```

Adding and Deleting Tuples

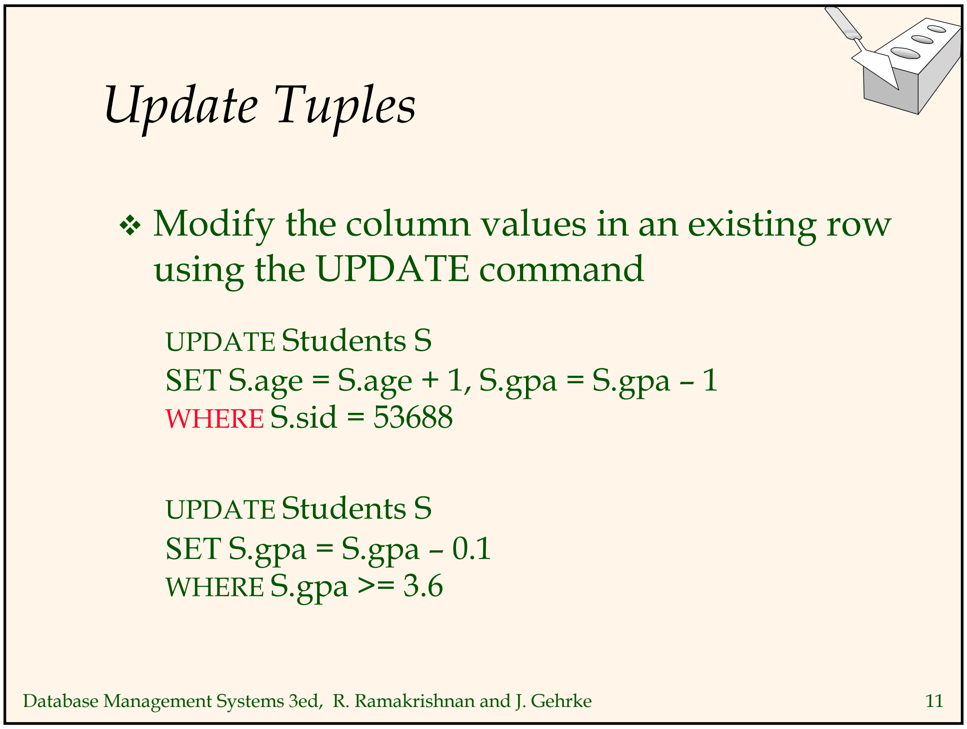
- Can insert a single tuple using:

```
INSERT INTO Students (sid, name, login, age, gpa)
VALUES (53688, ‘Smith’, ‘smith@cs’, 18, 3.2)
```

- Can delete all tuples satisfying some condition (e.g., name = Smith):

```
DELETE
FROM Students S
WHERE S.name = ‘Smith’
```

☛ *Powerful variants of these commands are available; more later!*

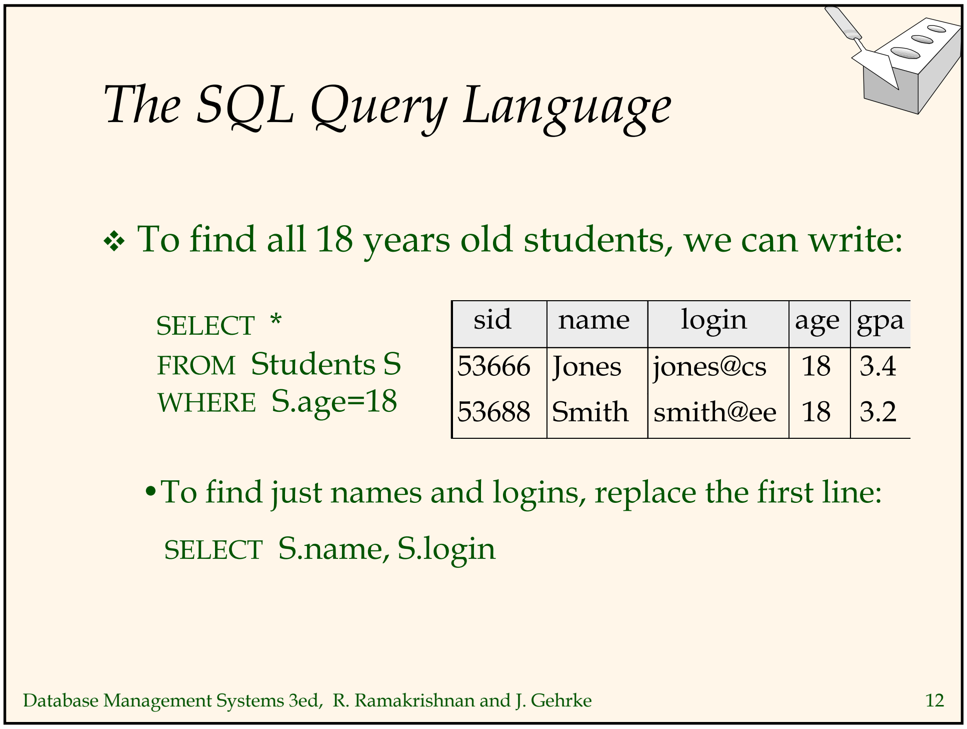


Update Tuples

- Modify the column values in an existing row using the UPDATE command

```
UPDATE Students S
SET S.age = S.age + 1, S.gpa = S.gpa – 1
WHERE S.sid = 53688
```

```
UPDATE Students S
SET S.gpa = S.gpa – 0.1
WHERE S.gpa >= 3.6
```



The SQL Query Language

- To find all 18 years old students, we can write:

```
SELECT  *
FROM  Students S
WHERE  S.age=18
```

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2

- To find just names and logins, replace the first line:

```
SELECT  S.name, S.login
```

Querying Multiple Relations

- What does the following query compute?

```
SELECT  S.name, E.cid
FROM  Students S, Enrolled E
WHERE  S.sid=E.sid AND E.grade="A"
```

Given the following instances of Enrolled and Students:

sid	cid	grade
53831	Carnatic101	C
53831	Reggae203	B
53650	Topology112	A
53666	History105	B

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@eecs	18	3.2
53650	Smith	smith@math	19	3.8

we get:

S.name	E.cid
Smith	Topology112

Destroying and Altering Relations

```
DROP TABLE  Students
```

- Destroys the relation Students. The schema information *and* the tuples are deleted.

```
ALTER TABLE  Students
        ADD COLUMN firstYear: integer
```

- The schema of Students is altered by adding a new field; every tuple in the current instance is extended with a *null* value in the new field.