CS 2451
Database Systems:
Relational Data Model

http://www.seas.gwu.edu/~bhagiweb/cs2541

Spring 2020

Instructor: Dr. Bhagi Narahari & R. Leontie

1

Today…

- Relational Data Model- Definitions
  - High level concepts
  - We discuss how these are implemented in SQL
    More details when we cover SQL
- Front end tools – HTML/CSS/…
- Next class - start with formal query languages before moving to SQL

2

Database "People" – i.e., roles

- There are several types of database 'personnel':
  - ***Database administrator*** (***DBA***) - responsible for installing, maintaining, and configuring the DBMS software.
    - *gets paid the big bucks !*
  - ***Data administrator*** (***DA***) - responsible for organizational policies on data creation, security, and planning.
    - *Gets to be on the hot seat if data leaks happen..*
  - ***Database designer*** - defines and implements a schema for a database and associated applications.
    - ***Logical/Conceptual database designer*** - interacts with users to determine data requirements, constraints, and business rules.
    - ***Physical database designer*** - implements the logical design for a data model on a DBMS. Defines indexes, security, and constraints.
  - ***DBMS developer*** - writes the DBMS software code.
  - ***Application developer*** - writes code that uses the DBMS.
  - ***User*** - uses the database directly or through applications.

YOU

3

Some Terminology

- **Database:**
  - A collection of related data.
- **Data:**
  - Known facts that can be recorded and have an implicit meaning.
- **Mini-world:**
  - Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.
- **Database Management System (DBMS):**
  - A software package/ system to facilitate the creation and maintenance of a computerized database.
    Example: MySQL, Oracle, MongoDB
- **Database System:**
  - The DBMS software together with the data and (usually) the applications

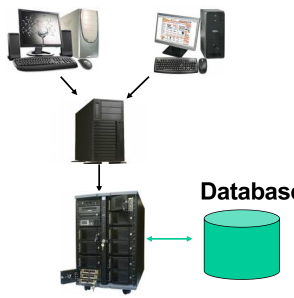
4

Recent Developments
(Recent in my timeline ☺)

- Social Networks started capturing a lot of information about people and their communications (tweets, photos, videos..)
  - \- Facebook, Twitter, Linked-In,….
- All of the above constitutes **data**
- Search Engines: Google, Bing, Yahoo : collect their own repository of web pages for searching purposes
- New Technologies emerging to manage vast amounts of data generated on the web:
  - Big Data storage systems involving large clusters
  - NOSQL (Not Only SQL) systems
  - A large amount of data now resides on the "cloud" which means it is in huge data centers using thousands of machines.

5

How will your database system be architected: Three-Tier Client-Server Architecture



Tier 1: Client (Web/mobile)
•User Interface (using HTML/CSS)

Tier 2: Application Server
•Business logic – written using PHP
•Data processing logic

Tier 3: Database Server
•Data storage/management
•Using MySQL

6

Data Models..more definitions

- **Data Model:**
  - A formal framework to describe the data and ***structure,*** **constraints**, of a database, and
  - the ***operations*** for manipulating these structures
  - Constructs used to define the database structure

    Data types, format (table),…ex: name is a char string
- **Constraints:**
  - Constraints specify some restrictions on valid data; these constraints must be enforced at all times..ex: GWID is unique
- **Data Model Operations:**
  - These operations are used for specifying database *retrievals* and *updates* by referring to the constructs of the data model.

    Ex: find student name with GWID= abcd

7

More terminology..Levels of Data Models

- **Conceptual (high-level, semantic) data models:**
  - Provide concepts that are close to the way many users perceive data.
- **Logical level:**
  - Provide concepts used by DBMS implementations
  - *In earlier definition we mixed conceptual and logical
- **Physical (low-level, internal) data models:**
  - details of how data is stored in the DBMS/computer.
  - These are usually specified in an ad-hoc manner through DBMS design and administration manuals
- **Self-Describing Data Models:**
  - Combine the description of data with the data values. Examples include JSON/XML, key-value stores and some NOSQL systems.

8

Conceptual Data Model: The Entity-Relationship (ER) Model

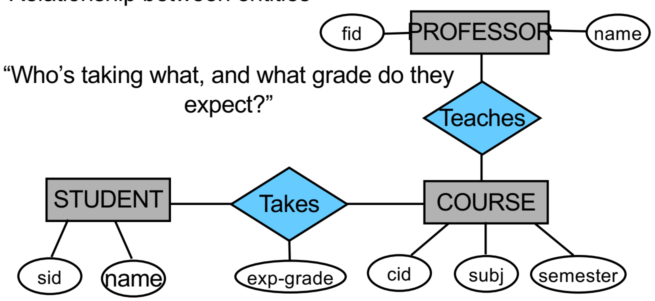
- Provide database design that is easy to interpret by a wide class of users
  - Not just database/CS experts
  - You want to provide a design to a “client” using a representation they can understand
- What’s a natural way to provide an easy to interpret representation….
  - Fill in the blanks: A ________ is worth a thousand words
- “Visual” representation of the data, how it interacts, constraints, etc.
  - And can be automatically mapped to a data model (relational)
- ER-Model is one such data model
  - We will return to it after we cover the relational model

9

Example: ER Design for mini-banner:

Entities = Professors, Students, Courses
Relationship between entities

“Who’s taking what, and what grade do they expect?”



One picture provides info on what your system stores and models
UML anyone ?

10

Some more terminology….. Schemas versus Instances

- *Similar to types and variables in programming languages*
- Database Schema: structure of the database
  - The ***description*** of a database.
  - Includes descriptions of the database structure, data types, and the constraints on the database.
  - Schema Diagram: ***illustrative*** display of a database schema.
- Database instance/state: actual data (content) stored in a database at a particular moment in time
  - Initial state: when database is loaded
  - Valid state: A state that satisfies the structures and constraints of the database…

    job of DBMS to ensure valid entries
  - The ***database schema*** changes very infrequently.

    Preferably never
  - The ***database state*** changes every time the database is updated.

11

Database Schema for a COMPANY Database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

12

Example of a Database Schema

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Populated database state/instance for COMPANY

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Example Instance & Schema

STUDENT

sid	name
1	Ross
2	Lee
3	Emily

Takes

sid	exp-grade	cid
1	A	550-0103
1	A	700-1003
3	C	500-0103

COURSE

cid	subj	sem
550-0103	DB	S13
700-1003	AI	S13
500-0103	Arch	F12

PROFESSOR

fid	name
1	Wood
2	Heller
8	Narahari

Teaches

fid	cid
1	550-0103
2	700-1003
8	500-0103

- Our focus now: relational schema – set of tables
- Can have other kinds of schemas – XML, object, …

History of Data Models

- Network Model
- Hierarchical Model
- Relational Model
- Object-oriented Data Models
  - Object-Relational Models
- NoSQL/ Big Data technologies
  - Schema-less designs

History of Data Models

- **Network Model:**
  - Network of records
  - The first network DBMS implemented by Honeywell in 1964-65
  - Can model complex relationships between records **i.e., graphs!**
- **Hierarchical Model**:
  - First implemented by IBM and Rockwell Intl. 1965 (?)
  - Hierarchical network of records (trees like)
  - Abstract view of an Org Chart – models hierarchy in the data
- **Query language**: COBOL
  - Hear about the Y2K bug/problem ?
- **Disadvantages:**
  - Database contains complex set of ***pointers*** that thread through sets of records

    Little scope for "query optimization", Procedural nature of processing

    Do you like working with pointers ? ☺

17

Next: Relational Model Concepts

- The relational Model of Data is based on the concept of a *Relation*
  - The strength of the relational approach to data management comes from the ***formal foundation provided by the theory of relations***
- start with review of essentials of the *formal relational model*
- In *practice*, there is a *standard model* based on SQL
- Note: There are several important differences between the *formal* model and the *practical* model
  - And, there are variations in SQL features provided on different DBMS systems

    Oracle SQL, MySQL, MS-SQL,…

18

Relational Model Concepts

- The model was introduced by Dr. E.F. Codd of IBM Research in 1970

  "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970

  Describes the data minimally and mathematically

  – A relation describes an association between data items – tuples with attributes

  Uses standard mathematical (logical) operations over the data – relational algebra or relational calculus
- It contributed: data independence, query languages, query optimization
- The above paper caused a major revolution in the field of database management and earned Dr. Codd the coveted ACM Turing Award

19

Why Did It Take So Many Years to Implement Relational Databases?

- Codd's original work: 1969-70
- Earliest relational database research: ~1976
- Commercial Relational DBMSs: ~mid 1980s
- Widespread deployment" mid-1990's
- Why the gap? Top 10 reasons…
  1. "You could do the same thing in other ways"
  2. "Nobody wants to write math formulas"
  3. "Why would I turn my data into tables?"
  4. "It won't perform well"
  5. …
- *What do you think?*

20

Relational Model Definitions

- A ***relation*** is a table with columns and rows.
- An ***attribute*** is a named column of a relation.
- A ***tuple*** is a row of a relation.
- A ***domain*** is a set of allowable values for one or more attributes.
- The ***degree*** of a relation is the number of attributes it contains.
- The ***cardinality*** of a relation is the number of tuples it contains.
- A ***relational database*** is a collection of normalized relations with distinct relation names.

21

Example of a Relation

Relation Name

Attributes

Tuples

STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 5.1
The attributes and tuples of a relation STUDENT.

Degree= 7, Cardinality = 5

22

Relational Model: Formal Definition

- Formally, a table is a relation over K sets (domains)
  - $R \subseteq A_1 \times A_2 \ldots. \times A_K$

    Subset of the cartesian product of the K domains
  - Tuple= $(t_1,t_2,\ldots,t_K)$, where $t_i \in A_i$
- A database is a collection of relations
- Theoretically: a relation is a set of tuples; no tuple can occur more than once
  - *Real systems may allow duplicates for efficiency or other reasons – we'll ignore this for now*

23

Relation Schemas and Instances

- A ***relation schema*** is a definition of a single relation.
- A ***database schema*** is a set of relation schemas (modeling a particular domain).
- A ***relation instance*** denoted ***r*(*R*)** over a relation schema ***R***$(A_1, A_2, \ldots, A_n)$ is subset of the *Cartesian product* of the *domains* of all *attributes* in the *relation schema*
  - **r(R)** $\subseteq dom(A_1) \times dom(A_2) \times \ldots \times dom(A_n)$
  - i.e., a set of n-tuples $<d_1, d_2, \ldots, d_n>$ where each d_i is an element of $dom(A_i)$ or is **null**.
  - A value of **null** represents a missing or unknown value.

24

Definition Summary

Informal Terms		Formal Terms
Table		Relation
Column Header		Attribute
All possible Column Values		Domain
Row		Tuple
Cardinality		Number of rows
Table Definition		Schema of a Relation
Populated Table		State of the Relation

25

Properties of relations

- Each relation name is unique
  - No two relations have the same name
- Each cell of the relation (value of a domain) contains exactly one atomic (single) value
- Each attribute of a relation has a distinct name
- values of an attribute are all from the same domain
- Each tuple is distinct. There are no duplicate tuples
  - Properties of a relation; in SQL can be bags (allow duplicates)
- order of attributes is not really important
  - Note difference from mathematical def of relations
    Tuple (x,y) is not the same as (y,x) in def of relation
  - Reason: attribute names represent domain and can be reordered
- Order of tuples is not important

26

Properties and constraints of the data

- Data represented as a relation/table
- Schema specifies the attributes/columns and their type
- Any other properties we need to define to capture the "application"
- Need to capture the 'business' rules:
  - How do we uniquely identify a student ?
  - Can their letter grade be any alphabet ?
  - Can a student take a course that is not in the course schedule/bulletin ?
  - ….
- Concept of **Constraints**

27

CONSTRAINTS

Constraints determine which values are permissible (or not) in the database. three main types:

- **Inherent or Implicit Constraints**:
  - These are based on the data model itself. (E.g., relational model does not allow a list as a value for any attribute)
- **Schema-based or Explicit Constraints – Integrity Constraints**: are rules or restrictions that apply to the database and limit the data values it may store
  - They are expressed in the schema by using the facilities provided by the model.
- **Application based or semantic constraints**:
  - These are beyond the expressive power of the model and must be specified and enforced by the application programs.

28

Integrity Constraints (ICs)

- IC: condition that must be true for *any* instance of the database; e.g., *domain constraints.*
  - ICs are specified when schema is defined.
  - ICs are checked when relations are modified.
- A *legal* instance of a relation is one that satisfies all specified ICs.
  - DBMS should not allow illegal instances.
- Why is this useful
  - If the DBMS checks ICs, stored data is more faithful to real-world meaning.
- *Think of the constraints as the business rules derived from the application*
- *Carefully analyze the semantics of the real world app before defining the constraints*

29

Where do ICs Come From?

- ICs are based upon the semantics of the real-world enterprise that is being described in the database relations.
  - *Need to carefully analyze the application before reaching a conclusion on the Integrity Constraints!*
- We can check a database instance to see if an IC is violated, but we can NEVER infer that an IC is true by looking at an instance.
  - An IC is a statement about *all possible* instances!
  - From example, we know *name* is not a key, but the **assertion** that *sid* is a key is given to us.
- Key and foreign key ICs are the most common; more general ICs supported too.
  - Shall return to these after we cover DML aspect of SQL

30

Definition: Relational Model Integrity Constraints

- Integrity rules are used to ensure the data is accurate.
- Types of **constraints**:
  - ***Domain constraint*** - Every value for an attribute must be an element of the attribute's domain or be **null**.

    null represents a value that is currently unknown or not applicable.

    null is not the same as zero or an empty string.
  - ***Entity integrity constraint*** - In a base relation, no attribute of a primary key can be null.
  - **Key constraint** – every relation must have a key
  - ***Referential integrity constraint*** - If a foreign key exists in a relation, then the foreign key value must match a primary key value of a tuple in the referenced relation or be null.

31

Key Constraints

- **Superkey** of R:
  - Is a set of attributes SK of R with the following condition:

    No two tuples in any valid relation state r(R) will have the same value for SK

    That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK]
    This condition must hold in *any valid state* r(R)
- Candidate **Key** of R:
  - A "minimal" superkey
  - That is, a key is a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey (does not possess the superkey uniqueness property)
  - A Key is a Superkey but not vice versa
- **Primary Key**: a DBA chosen key for the relation/table

32

Primary Key Constraints

- *Every relation must have a key*
- A set of fields is a *key* for a relation if :
  1. No two distinct tuples can have same values in all key fields, and
  2. This is not true for any subset of the key.
  - Part 2 false? A *superkey*.
  - If there's >1 key for a relation, one of the keys is chosen (by DBA) to be the *primary key*.
- E.g., what is a key for Students relation ?
- *sid* is a key for Students. (What about *name*?) The set {*sid, gpa*} is a superkey.

33

Key Constraints (continued)

- Example: Consider the CAR relation schema:
  - CAR(State, Reg#, SerialNo, Make, Model, Year)
  - CAR has two keys:
    - Key1 = {State, Reg#}
    - Key2 = {SerialNo}
  - Both are also superkeys of CAR
  - {SerialNo, Make} is a superkey but *not* a key.
- In general:
  - Any *key* is a *superkey* (but not vice versa)
  - Any set of attributes that *includes a key* is a *superkey*
  - A *minimal* superkey is also a key
  - Can have many candidate keys, one of them chosen as primary key

34

Key Constraints (continued)

- If a relation has several **candidate keys**, one is chosen arbitrarily to be the **primary key**.
  - The primary key attributes are underlined.
- Example: Consider the CAR relation schema:
  - CAR(State, Reg#, SerialNo, Make, Model, Year)
  - We chose SerialNo as the primary key
- The primary key value is used to *uniquely identify* each tuple in a relation
  - Provides the tuple identity
- Also used to *reference* the tuple from another tuple
  - General rule: Choose as primary key the smallest of the candidate keys (in terms of size)
  - Not always applicable – choice is sometimes subjective

35

Referential Integrity

- A constraint involving **two** relations
  - The previous constraints involve a single relation.
- Used to specify a **relationship** among tuples in two relations:
  - The **referencing relation** and the **referenced relation**.
- Examples:
  - Student enrolled in a course
  - Employee working on a project: in Works_On table

36

Foreign Keys in SQL

- Only students listed in the Students relation should be allowed to enroll for courses.
  - If a value of sid appears in Enrolled relation then it MUST appear in Students relation
    "Only students can take courses"

Enrolled

sid	cid	grade
53666	Jazz101	C
53666	Reggae203	B
53650	Topology112	A
53666	History105	B

Students

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

37

Foreign Keys, Referential Integrity

- *Foreign key* : Set of fields in one relation that is used to `refer' to a tuple in another relation. (Must correspond to primary key of the second relation.)
  - Like a `logical pointer'.
- In *Enrolled* table – *sid* is a student, what can we say about the *students* table ?
  - Enrolled(*sid*: string, *cid*: string, *grade*: string)
- *sid* in enrolled is a foreign key referring to Students:
  - The student with this ID MUST exist in the Students table
  - If all foreign key constraints are enforced, *referential integrity* is achieved, i.e., no dangling references.

38

Referential Integrity (or foreign key) Constraint – more general definition

- Statement of the constraint
  - The value in the foreign key column (or columns) FK of the **referencing relation** R1 can be **either**:
    (1) a value of an existing primary key value of a corresponding primary key PK in the **referenced relation** R2, or
    (2) a **null**.
- In case (2), the FK in R1 should **not** be a part of its own primary key.

39

Other Types of Constraints

- Semantic Integrity Constraints:
  - based on application semantics and cannot be expressed by the model per se
  - Example:
    - "the max. no. of hours per employee for all projects he or she works on is 40 hrs per week";
    - Grade cannot be any alphabet; …
- A **constraint specification** language may have to be used to express these
- SQL-99 allows **CREATE TRIGGER** and **CREATE ASSERTION** to express some of these semantic constraints
- Keys, Permissibility of Null values, Candidate Keys (Unique in SQL), Foreign Keys, Referential Integrity etc. are expressed by the **CREATE TABLE** statement in SQL.

40

When do Integrity Constraints get triggered…

41

COMPANY Database Schema

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Figure 5.5
Schema diagram for the COMPANY relational database schema.

Slide 5- 42

42

Populated database state

- Whenever the database is changed, a new state arises
- Basic operations for changing the database:
  - INSERT a new tuple in a relation
  - DELETE an existing tuple from a relation
  - MODIFY an attribute of an existing tuple
- Next slide shows an example state for the COMPANY database schema
- The update operations must keep the database in a consistent state – i.e. all instances must satisfy integrity constraints
- Updates may **propagate** to cause other updates automatically. This may be necessary to maintain integrity constraints.

43

Populated database state for COMPANY

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

44

Update Operations on Relations

- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (RESTRICT or REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine

45

Possible violations for each operation

- INSERT may violate any of the constraints:
  - Domain constraint:

    if one of the attribute values provided for the new tuple is not of the specified attribute domain
  - Key constraint:

    if the value of a key attribute in the new tuple already exists in another tuple in the relation
  - Referential integrity:

    if a foreign key value in the new tuple references a primary key value that does not exist in the referenced relation
  - Entity integrity:

    if the primary key value is null in the new tuple

46

Possible violations for each operation

- DELETE may violate only referential integrity:
  - If the primary key value of the tuple being deleted is referenced from other tuples in the database

    Can be remedied by several actions: RESTRICT, CASCADE, SET NULL (see Chapter 6 for more details)
    - RESTRICT option: reject the deletion
    - CASCADE option: propagate the new primary key value into the foreign keys of the referencing tuples
    - SET NULL option: set the foreign keys of the referencing tuples to NULL
  - One of the above options must be specified during database design for each foreign key constraint

47

Possible violations for each operation

- UPDATE may violate domain constraint and NOT NULL constraint on an attribute being modified
- Any of the other constraints may also be violated, depending on the attribute being updated:
  - Updating the primary key (PK):

    Similar to a DELETE followed by an INSERT

    Need to specify similar options to DELETE
  - Updating a foreign key (FK):

    May violate referential integrity
  - Updating an ordinary attribute (neither PK nor FK):

    Can only violate domain constraints

48

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.

49

Relational Query Languages

- *Query languages:*
  - Allow specification of schemas and constraints
  - Allow manipulation and retrieval of data from a database.
- Relational model supports simple, powerful QLs:
  - Strong formal foundation based on logic.
  - Allows for much optimization.
- Query Languages **!=** programming languages!
  - QLs not expected to be "Turing complete".
  - QLs not intended to be used for complex calculations.
  - QLs support easy, efficient access to large data sets.

50

Formal Query Languages

- Formal query languages are defined as mathematical operators over the set
  - advantage of a formal language ?
  - Relational algebra, Relational calculus are examples
- Procedural vs Non-procedural languages
  - Procedural: what data to fetch from DB and how/where to get the data
  - Non-procedural: what data to fetch from DB
    - System/DBMS needs to figure out the "how"
  - Can have a mix in practice
  - **Relational algebra**: procedural language
  - **Relational calculus**: non-procedural (declarative)

51

SQL: Structured Query Language

The standard language for relational data
- Invented by folks at IBM, esp. Don Chamberlin
- Actually not a great language…
- Beat a more elegant competing standard, QUEL, from Berkeley

Separated into a DML & DDL

SQL DML component based on relational algebra & calculus

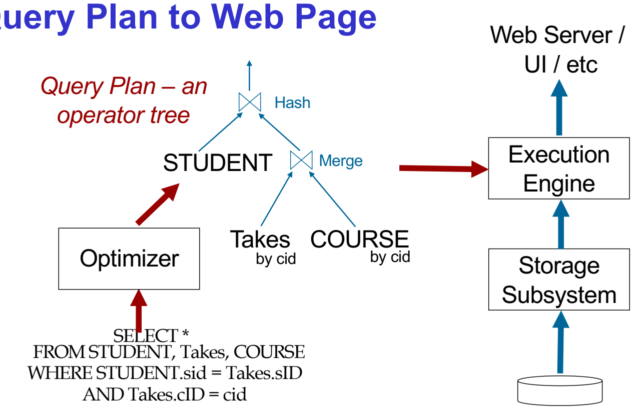
52

SQL

- components
  - Data definition (DDL) – to define schema/tables
    - Define Schema
    - Define Constraints
  - Manipulation/query (DML) – for queries
  - Transaction control – to specify a transaction
  - Index – to specify storage and indexing schemes
  - Authorization- for access control/security
    - We will cover the DDL and query part of SQL first
    - Shall return to the other components after we cover those topics

53

The Big Picture: SQL to Algebra to Query Plan to Web Page



54