Relational Model and Relational Algebra

Rose-Hulman Institute of Technology
Curt Clifton

Administrative Notes

- Grading Weights
- Schedule Updated

Review – ER Design Techniques

- Avoid redundancy and don’t duplicate data
- Don’t use entity set when attribute will do
- Limit use of weak entity sets

Review – Relations

- Formally
  - Tuple: an ordered list
    - Each value drawn from some domain
  - *n*-tuple: an ordered list of length *n*
  - Relation: a **set** of *n*-tuples
- Informally:
  - Relation: a table with unique rows
  - Rows = tuples; Columns = attributes;
  - Values in column = domain
- *Database*: a collection of relations

Review – Schemas

- Relation schema
  - Describes a relation
  - *RelationName (AttrName1, AttrName2,…)*
  - Or *RelationName (AttrName1:type1, …)*
- Database schema
  - Set of all the relation schema for the DB's relations

Review – Converting ER Diagrams

- Entity sets become relations
  - Columns are attributes of entity set
- Relationships also become relations
  - Columns are keys of participating entity sets
- Can avoid relations for many-one relationships
  - Add key of the one to the relation of the many

Relational Model

- Structure – sets of *n*-tuples
- Basic Operations – the relational algebra
  - Set Union, Intersection, and Difference
  - Selection
  - Projection
  - Join

More On Structure

- Let $R(A1, A2, \ldots, An)$ be a relation schema
- For each tuple of the relation R…
  - Its ith element comes from **domain** of Ai
- Write "$r(R)$" for a **value** of R
  - $r(R) \subseteq \mathrm{dom}(A1) \times \mathrm{dom}(A2) \times \ldots \times \mathrm{dom}(An)$
- Write $t \in r(R)$ for a tuple in R
- Write $t[K]$ for subtuple of t,
  where K is a set of attribute names

Relational Integrity Constraints

- Conditions that must hold for a relation instance to be **valid**
- Two main types
  - Entity Integrity
  - Referential Integrity
- Need a few more terms before we can define these…

Keys, Formally

- Some terms:
  - **Superkey** of *R*: set of attributes *SK* of *R* such that no two tuples in any valid instance *r(R)* have the same value for *SK*
  - **Key** of *R*: a minimal superkey *K*
    - Remove any attribute from *K* and it's no longer a superkey
  - **Candidate key**: any one of several keys
  - **Primary key**: the chosen key, *PK*, for the relation

Entity Integrity Defined

- Let *DB* be a database schema
  - $DB = \{R1, R2, \ldots, Rn\}$
  - Where each *Ri* is a relation schema
- **Entity integrity**: for every tuple *t* in every relation *Ri* of *DB*, $t[PKi] \neq$ null, where *PKi* is the primary key of *Ri*
  - Primary keys can't be null!

Foreign Keys

- Specify **relationship** between tuples in different relations
- Referencing relation, *R1*, has foreign key attributes *FK*
- Referenced relation, *R2*, has primary key attributes *PK*
- For $t1 \in R1$, $t1[FK] = t2[PK]$ for some $t2 \in R2$
- Shown with arrows…

Example – Foreign Keys

- Easy to identify foreign keys when converting from ER Diagram, they encode relationships
- Can also find them in relation schemas

Referential Integrity Defined

- The value of the foreign key of a referencing relation can be either:
  - the value of an existing primary key in the referenced relation, **or**
  - null

Relational Model

- Structure – sets of *n*-tuples satisfying
  - Entity Integrity
  - Referential Integrity
- Basic Operations – the relational algebra
  - Set Union, Intersection, and Difference
  - Selection
  - Projection
  - Join

What is an "Algebra"?

- Name from Muhammad ibn Musa al-Khwarizmi's (780–850) book *al-jabr*
  - About arithmetic of variables
- An Algebra includes
  - Operands – variables or values
  - Operators – symbols denoting operations

Relational Algebra

- A formal model for SQL
- Operands
  - Relations
  - Variables
- Operators
  - Formal analogues of DB operations

Basic Set Operators

- Intersection
  - $R1 \cap R2$
  - All tuples that are in both *R1* and *R2*
- Union
  - $R1 \cup R2$
  - Any tuple that is in either *R1* or *R2* (or both)
- Difference
  - $R1 \setminus R2$
  - All tuples that are in *R1* but are not in *R2*
- *R1* and *R2* must be **compatible** – attribute types match

Selection, σ

- **For picking rows** out of a relation
- $R1 \leftarrow \sigma_C(R2)$
  - *C* is a boolean condition
  - *R1* and *R2* are relations
  - *R1* **gets** every tuple of *R2* that satisfies *C*
- Selection is commutative
  - $\sigma_{C1}(\sigma_{C2}(R2)) = \sigma_{C2}(\sigma_{C1}(R2)) = \sigma_{C1 \wedge C2}(R2)$

Projection, π

- **For picking columns** out of a relation
- $R1 \leftarrow \pi_L(R2)$
  - L is a list of attribute names from $R2$'s schema
  - $R1$ and $R2$ are relations
  - Attributes of $R1$ are given by L
  - $R1$ **gets** every tuple of $R2$ but just attributes from L
- Is Projection commutative?
  - $\pi_{L1}(\pi_{L2}(R2)) =? \pi_{L2}(\pi_{L1}(R2))$

Product

- **Combining tables** *without* matching
- $R \leftarrow R1 \times R2$
  - *R1* and *R2* are relations
  - Pair every tuple from *R1* with every tuple from *R2*
  - *R* gets every attribute of *R1* and every attribute of *R2*
    - Can use *R1.A* naming convention to avoid collisions
- If *R1* has 10 rows and *R2* has 42, how many in *R?*

Theta-Join

- **Combining tables** *with* matching
- $R \leftarrow R1 \bowtie_C R2$
  - *R1* and *R2* are relations
  - *C* is a boolean expression over attributes of *R1* and *R2*
  - Pair every tuple from *R1* with every tuple from *R2* **where** *C* **is true**
  - *R* gets every attribute of *R1* and every attribute of *R2*
  - $R1 \bowtie_C R2 = \sigma_C(R1 \times R2)$
- If *R1* has 10 rows and *R2* has 42, how many in *R?*

Equijoin

- A theta-join using an equality comparison
- Really just a $5 word, but you might see it

Natural Join

- Joins two relations by:
  - Equating attributes of the same name
  - Projecting out one copy of each shared attribute
- $R \leftarrow R1 * R2$

Dangling Tuple Problem

- Suppose DEPT_LOCATION had no entry for Houston
- Consider:
  - *R* ← *DEPARTMENT* * *DEPT_LOCATIONS*
- What happens to Headquarters?

Outer Joins

- Solve the *dangling tuple* problem
- If a tuple would be dropped by the join, then include it and use null for the other attributes
- Shown as a bow tie with “wings”
  - Wings point to relation whose tuples **must** appear

Renaming

- $\rho_{R1(A1,\ldots,An)}(R2)$
  - Rename *R2* to *R1*
  - Rename attributes to *A1, …, An*
- Usually just play fast and loose:
  - $R1 \leftarrow \rho_{A1,\ldots,An}(R2)$, or
  - $R1(A1, \ldots, An) \leftarrow R2$

Combining Expressions

- Nesting:
  - $R \leftarrow \pi_L(\sigma_C(R1 \times R2))$
  - Work inside out like you're used to
- Sequencing:
  - $Rc \leftarrow R1 \times R2$
    $Rs \leftarrow \sigma_C(Rc))$
    $R \leftarrow \pi_L(Rs)$

Homework Problem 6.18

- Parts a–d and g
- Begin in class, may work in groups of 2–3
  - Please note your partners on the sheet