1 Overview

- **tuple**: an ordered and named collection of values
  Ex: `('Mehta, Dinesh', 'CSCI 406', 'ALGORITHMS')`
- **attributes**: formally, the attributes are a set
  Ex: `('instructor', 'course-id', 'title')`
- **domain**: each attribute has associated domain from which values come
- **relation schema**:
  - A relation schema R is denoted as $R(A_1, A_2, \ldots, A_n)$.
  - Has **degree** n
  - Attributes A_i
  - Each attribute has associated domain: $Dom(A_i)$
- **relation or relation state**:
  - A relation is a set of tuples $r(R)$ conforming to the relation schema: for any tuple $t \in r$

    $$t = (v_1, v_2, \ldots, v_n) : v_i \in Dom(A_i)$$

  - In other words: $r(R) \subseteq (dom(A_1) \times dom(A_2) \times \cdots \times dom(A_n))$
  - **In actuality** real RDBMS
    * tables may contain duplicates
    * no ordering of tuples
- `NULL`
  A "value" that can exist in a relational database representing
  - a value is unknown
  - value is irrelevant
  - value is missing

  problems with `NULL`
  - `NULL` values cannot be compared

    $$a = b, a \text{ } is \text{ } NULL, b \text{ } is \text{ } NULL \rightarrow false$$

    instead we have `ISNULL` operator
- **constraints**
  - on the relation schema
  - between relation schemas
- **superkey** - A superkey of R is a subset of R's attributes such that no relation $r(R)$ may contain tuples with exactly the same values for attributes in the superkey.
  - any superset of a superkey is also a superkey
  - every relation (in theory) has at least one superkey - the set of all attributes of the schema
- **Key** - A superkey s.t. no attribute can be removed from the key without destroying the superkey property: a *minimal* superkey
- **Candidate Keys** - All the keys of a schema are called *candidate keys*
- **Primary Key** - By convention, we choose one candidate key as the *primary key*
  - we tend to prefer "smaller" keys (e.g. `crn`)
  - in practice in SQL DB:
    * primary keys impose uniqueness constraint on data
    * other candidate keys → impose uniqueness constraint
- **Referential Integrity/Foreign Key Constraint** Constraint on the relationship between relations. A *foreign key* is a subset of attributes of a relation schema with the property that its values are either `NULL` or existing in the specified attributes of a referenced table.

2 Relational Theory

- each relation is a **set** of tuples
- tuple = set of values associated with named attributes
- **Relational Algebra**
  - useful operations that can be applied to relations
  - resulting algebra

3 Unary Operations $-2+1$

3.1 Selection

- Chooses subsets of tuples from a relation according to some condition (`WHERE`)
- Letting R represent some relation: $\sigma_{\text{condition}}(R)$
- E.g. $\sigma_{\texttt{course\_id = 'CSCI 403'}}(\texttt{mines\_courses})$
- Can use `AND`, `OR`, `NOT`, etc.
- Properties of σ
  - degree of relation (# of attributes) is the same as the original
  - the number of tuples in the result $\leq$ # tuples in original
  - commutative $\sigma_a(\sigma_b(R)) = \sigma_b(\sigma_a(R))$
  - any sequence of σ operations can be replaced with a single σ (using `AND`)

3.2 Projection

- chooses attributes from set of attributes in a relation
- parallel operation in SQL: `SELECT`
- Notated:

  $$\pi_{attr_1, attr_2, \ldots}(R)$$

- Example:

  $$\pi_{instructor, course\_id}(mines\_courses)$$

- Properties of π
  - # of tuples is always $\leq$ # tuples in original
  - **not** commutative: rather

    $$\pi_{list_1}(\pi_{list_2}(R)) = \pi_{list_1}(R)$$

    if $list_1 \subset list_2$
    else, ill formed

3.3 Renaming - ρ

- parallel operation in SQL: `AS`
- Notation:

  $$\rho_{S(B_1, B_2, \ldots)}(R)$$

- Example: Let table X have attributes (a, b, c).

  $$\rho_{Y(d,e,f)}(X)$$

```
SELECT a AS d,
       b AS e,
       c AS f
  FROM X as Y;
```

3.4 Sequences of Operations

3.4.1 Representations

1. nesting: example

$$\rho_{(name, courseid)}(\pi_{instructor, courseid}(\sigma_{department='CS'}(mines\_cour$$

```sql
SELECT instructor AS name
       course_id
  FROM mines_courses
 WHERE department = 'CS';
```

2. sequence of named relations

$$R_1 = \sigma_{department='CS'}(mines\_courses)$$
$$R_2 = \pi_{instructor,course\_id}(R_1)$$
$$R_3 = \rho_{(name,course\_id)}(R_2)$$

4 Binary Operations

4.1 Set Operators

- $A \cup B$ - union
- $A \cap B$ - intersection
- $A - B$ - difference

4.1.1 Properties of Set Operators

- Union, intersection are commutative and associative
- Set difference is not commutative or associative

4.2 Cartesian Product and Joins

- $A \times B$ - pairs every tuple from A with every tuple from B
- If A has m attributes, B has n, $A \times B$ has $m + n$ attributes $|A \times B| = |A| \times |B|$.
- Typical Usage:

$$\sigma_{condition}(A \times B) \equiv A \bowtie_{condition} B$$

- Example: $mines\_courses \bowtie_{instructor=name} mines\_eecs\_faculty$

4.3 Theta Joins

- When:
  - we have $A \bowtie_{cond} B$
  - condition is of general form $cond_1 AND cond_2 AND \ldots$
  - each $cond_n$ $A_i \Theta B_j, A_i \in A, B_i \in B$

  Then we call $A \bowtie_{cond}$ a *theta join*: $A \bowtie_{\Theta} B$
- Further, when Θ is $=$, then $A \bowtie_{\Theta} B$ is called and *equijoin* which implies that there is some duplicate data column.
- If $A \bowtie_{cond} B$ is an equijoin and condition equates attributes of A with attributes of B of the **same name**, then the join is a *natural join*.
  $A * B$ - books notation
  $A \bowtie B$ - other people's notation
  Example: both `mines_courses` and `mines_courses_meetings` have `crn` field. Then $mines\_courses * mines\_courses\_meetings$. (Note, this automatically projects away duplicate column(s).)

5 Completeness

You can demonstrate that: $\sigma, \pi, \cup, -, \times$ is a complete set of operators for relational algebra.

6 Odds and Ends

- Division: $\div$ ~inverse of $\times$ (no mirror in SQL)
- Aggregates and grouping: not part of basic algebra