Database Systems

Relational Data Model

References:

1. Database Systems the Complete Book. Garcia-Molina, Ullman, Widom

Contents

- Data models
  - What is it?
  - Overview of different kinds of data models.
- Relational model, and,
- a portion of SQL
  - Used to define relations and their structure.
- Introduction to Relational Algebra.
- SQL serves as both
  - Query language: enables us to ask questions about the data.
  - Constraint language: enables us to restrict the data in the database in various ways.

Overview of Data Models

- Data model is very fundamental in the study of database systems.
- A data model is a notation for describing data or information. It typically has three parts:
  1. Structure of data.
  2. Operations on the data.
  3. Constraints on the data.

Data models: Structure of Data

- In programming languages e.g., C or Java, the structure of the data used by the program is often defined using
  - Arrays, structs (or objects), arrays of structs, structs containing arrays, etc.
- Data models are at a higher level than data structures.
- Typically allow high level specification of data (usually do not specify the detailed data structure).

Data Models: Data Operations

- In programming languages, anything that is programmable is a data operation.
- In database models, there is usually a limited set of data operations permitted.
- A limited set of queries (operations to retrieve information) and modifications (operations that change the database).
- Specification of queries and modifications is at a higher level than typical programming languages.
  - Easy to specify queries.
  - Brief.
- Yet, implementation of the queries is fast, and done by the database system.
  - Query optimization.
    - Contrast: Replacing bubblesort by mergesort in C is impossible for a compiler.

Some important data models

- Relational model.
  - Object relational model.
- Object Oriented model
- Semi-structured model.
  - XML database.
  - Others…

Relational Model: Very High Level

- Based on tables. E.g.. A movie table.
- ***Movies***

title	year	length	genre
Sholay	1975	198	action
Chak De! India	2007	153	sports
3 Idiots	2009	171	comedy

- Movie table, each row describes one movie: their title, year of production, length in minutes and genre of the movie.
  - For e.g., table includes 3 data rows on 3 movies, in general, there can be many

Movies

title	year	length	genre
Sholay	1975	198	action
Chak De! India	2007	153	sports
3 Idiots	2009	171	comedy

- Each row in the table looks like a struct in C.
- The table is a collection of rows, which is like an array of structs in C.
- Each column has a header name: it names that field. E.g.
  - Movies.title, Movies.genre….
- Operations associated with tables in the relational model form the relational algebra.
  - E.g., Give all rows in the Movies table whose genre is comedy.
- Constraint portion of the relational model:
  - E.g., There may be only a fixed list of genres for the movies. Last column must take a value from this list.
  - E.g.:{title, year} pair forms a key. That is, there cannot be two different rows with the same title and produced in the same year.

Relational Model: Basics

Relational Schemas and Tuples

- We take the standard ordering of a relation schema to be the order used in its definition.
- Relational model has one or more relations.
  - Each relation has a schema.
  - The set of schemas for the relations of a database is called a relational database schema, or, database schema.
- Rows of a relation, other than the header row containing the attribute names, are called tuples. E.g.,
  1. (Gone with the Wind, 1939, 231, drama)
  2. (Star Wars, 1977, 124, ‘sciFi’), etc.
- Attributes of each tuple follow standard ordering.
- Attribute values are separated by commas.

Domains of Attributes

- Relational tuple requires each component of each tuple (i.e., attribute) to be atomic.
- i.e., elementary types, e.g., integer or string.
- But not structure, set, list, array, or any other complex type.
- Values associated with a relation attribute take values from a domain
  - Domain: an elementary type with possible NULL values. E.g.,
    - Movies(title, year, length, genre)
      1. title is of type string and is written as title:string.
      2. year: integer.
      3. length:integer.
      4. genre:string.
  - Movies(title:string, year:integer, length:integer, genre:string)

Equivalent representation of a Relation

- Relations are sets of tuples, not lists of tuples (i.e., no ordering).
- Order in the tuples of a relation are presented is immaterial.
- The three tuples of Movies relation can be equivalently represented in any of the 3! = 6 reorderings.

title	year	length	genre
Gone with the wind	1939	231	drama
Star Wars	1977	124	sciFi
Wayne's world	1992	95	comedy

- All the orderings is the same Movies relation.
- Moreover, attributes can be reordered arbitrarily. E.g.,

year	genre	title	length
1977	sciFi	Star Wars	124
1992	comedy	Wayne's World	95
1939	drama	Gone with the Wind	231

Relation Keys

- Key constraint:
  - A set of attributes forms a key for a relation if
    - we do not allow two tuples in a relation instance to have the same values in all the attributes of the key.
  - Indicate attribute(s) that form a key by underlining them.
    - Movies(title, year, length, genre)
    - Or, key is {title, year}.
    - That is, no two movies with the same title are produced in the same year.
  - A set of attributes forming a key is a statement about all possible instances of the relation that can occur in the database.
    - Not just about a single instance.
  - E.g., genre is not a key, since there can be many comedy movies, and many sciFi movies, etc.

Keys of Relations

- Many real-world databases use artificial keys,
  - even though there may be some attribute sets that can serve as a key.

- E.g., University may assign a unique RollNo to each student, even though, Aadhar No. of student may also serve as key.

  - Student(RollNo, Name, AadharNo, Address)

Example Schema

Movies(title:string, year:integer,
length:integer, genre:string,
studioName:string, producer#:integer)

- Movies schema, as earlier + two new attributes:
  - studioName gives the name of the studio that owns the movie.
  - producerC# is an integer giving an id to the producer of the movie (key of the MovieExec relation).
  - Key is {title, year}.

MovieStar(name:string, address:string,
gender:char,birthdate:date)

- This tells us about stars of movies.
- Key is name, the name of the moviestar.
  - No two movie stars assume the same name.
- Gender name is a single character M or F.
- Birthdate is of type date—a character string in a special form.

Example Schema-contd.

StarsIn(movieTitle:string,
movieYear:integer,
starName:string)

- This relation connects movies to the stars of that movie.
- A movie is defined by the pair of values in (movieTitle, movieYear) attribute pairs. Its details are in relation Movies.
- Star is identified uniquely by starName, details about the star is found in relation MovieStar.

MovieExec(name:string, address:string,
cert#:integer, netWorth:integer)

- This relation stores some info about movie executives (producer, director etc.)
- It contains their names, address and networth as data.
- cert# is a certificate number--- it is a key, Different movie executives have different cert#.

Example schema – closing

Studio(name:string,address:string,
presC#:integer)

- This relation stores basic info about movie studios that produce movies.
- Assumes that no two studios have the same name.
- presC# is the certificate number of the president of the studio, thereby identifying the president of the studio from the relation MovieExec.

Defining Relation Schema in SQL

- SQL (pronounced ``sequel'') is the principal database language.
- Current standard is SQL-99.
- Most commercial databases implement a close approximation to it.
- Two aspects of SQL:
  - Data-Definition Language: for declaring database schemas.
  - Data-Manipulation Language for
    - Querying (asking questions) about databases
    - Modifying the database (insert, delete, update data items)

Relations in SQL

- SQL distinguishes between three kinds of relations:
  - *Tables*: they are stored relations in the database.
    - Can be queried, modified.
  - Views: these are relations defined by computation.
    - Views are not stored.
    - Constructed in whole or in part, when needed.
  - Temporary tables:
    - constructed by the query processor when it is executing queries and data modifications.
    - They are not stored once the computation is over.

SQL Data Types

1. CHAR(n) and VARCHAR(n).
   - CHAR(n) denotes a fixed length string of up to n characters.
   - VARCHAR(n) also denotes a string of up to n characters.
   - Difference is implementation dependent.
     - CHAR(n) strings are padded to make n characters.
       - Padding character is typically the blank character.
       - E.g., CHAR(7) string: ‘hello’ is stored as ‘hello ‘.
     - VARCHAR(n) strings have endmarker or uses string length.

SQL Data Types: Bit(n), BOOLEAN, INT

- BIT(n), BITVARYING(n) type is bit strings of fixed or varying length.
- BIT(n) and BIT VARYING(n)
  - BIT(n) denotes bit strings of length n.
  - BIT VARYING(n) denotes bit strings of length up to n.
- Type BOOLEAN takes one of three logical values:
  - TRUE, FALSE or UNKNOWN.
- Type INT or INTEGER (they are synonyms) denotes integer values.
  - Type SHORTINT denotes short integers.
  - Similar to types int and short int in C and is implementation dependent.

Data types in SQL: Floating point no.

- FLOAT or REAL (they are synonyms) represent floating point numbers.
- DOUBLE gives higher precision.
  - Distinction between them is much like the C language.
- DECIMAL(n,d) also represents real numbers:
  1. Number has n decimal digits.
  2. d is the number of digits after decimal point.
  - E.g, 0123.45 is a possible value of type DECIMAL(6,2).
- NUMERIC is almost a synonym for DECIMAL.

SQL Data Types: Date and Time

- SQL standard: DATE type represents date in the format
  - '2022-01-10' represents 10 January 2022.
    - First four characters are digits representing the year.
    - Then a hypen and two digits representing month.
    - Then another hypen and two digits representing day.
    - Single digit month or day is padded with a leading 0.
- TIME type is a quoted string:
  - '14:01:2' represents the time 1 minute and 2 seconds after 2 pm.
    - Two digits for the hour, then a colon,
    - then, another two digits for the minute, then a colon,
    - and two digits for the second.
    - Fractional second is allowed, for e.g., '14:01:2.56' represents the time 1 minute, 2.56 seconds after 2pm.

SQL Table Declarations

- CREATE TABLE: Simple form of declaring a table followed by
  - Name of the relation and
  - attribute name list, within parenthesis and comma separated.
- CREATE TABLE Movies (

```
        title           VARCHAR(100),   /* title is declared as a string of
                                           upto 100 characters.
        year            INT,            year, length and producerC# are
                                           integers.
        length          INT,
        genre           CHAR(10),       10 characters are assumed to be
                                           enough to represent genre.
        studioName      CHAR(30),       30 characters are considered
                                           sufficient for studio name.
        producerC#      INT             */
);
```

CREATE TABLE contd.

```
CREATE TABLE MovieStar(
    name            CHAR(30),
    address         VARCHAR(255),
    gender          CHAR(1),
    birthdate       DATE
);
```

1. This is the MovieStar table.
2. Name and address are character strings of sizes up to 30 and 255.
3. gender attribute has values that are a single letter M or F.
4. Birthdate is of type birthdate.

Modifying Relation Schemas

- Imagine requiring to change the relation schema after
  - it has been in use for a long time, and
  - has many records (tuples) in the current instance.
- We might require to remove the entire table, e.g.,
  - Suppose there is a relation R.

    DROP TABLE R
  - Relation R no longer remains part of the database schema.

Modifying Relation Schemas

- If we need to modify the schema of an existing relation,
  - Use command ALTER TABLE
- To add an attribute name use
  - ADD followed by an attribute name and its data type.

    ALTER TABLE MovieStar ADD phone CHAR(16);
  - MovieStar now has an additional phone attribute which is a fixed length string of size 16 bytes.
  - *The value of the phone attribute in all tuples is initialized to* NULL.
    - This is in absence of *default value* (next slide).
- To drop an attribute name use
  - DROP followed by an attribute name.

    ALTER TABLE MovieStar DROP birthdate;

    deletes the birthdate attribute from MovieStar.

Default Value

- When tuples are created or modified, we may not have values for all its component attributes.
- There is a way to specify default values during attribute definition.
- E.g.,

  gender CHAR(1) DEFAULT '?'

  - Says that if gender attribute of a tuple is unspecified, then its value equals '?' .
- E.g.,

  birthdate DATE DEFAULT '0000-00-00'

  - Unspecified birthdate attribute gets the value day 0, month 0, year 0.

Declaring Keys in SQL

- E.g., Consider the table MovieStar.

  MovieStar(name, address, gender, birthdate)

- The attribute name is a **key** for the table MovieStar meaning,
  - No two tuples (records) in MovieStar have the same value for name.
  - Equivalently, each tuple in MovieStar table has a unique value in the name attribute.
- This can be specified using PRIMARY KEY clause in CREATE TABLE stmt.

```
CREATE TABLE MovieStar(
        name            CHAR(30),
        address         VARCHAR(255),
        gender          CHAR(1),
        birthdate       DATE,
        PRIMARY KEY     (name)   /* primary key specification*/    );
```

- Can also write instead of PRIMARY KEY specification

  UNIQUE (name) /* specifies that name is a unique key*/

Defining Keys

E.g., in the table Movies with schema

Movies(title, year, length, genre, studioName, producerC#)

- we assumed that the pair {title, year} forms a key. Meaning,
  - In a given year, there are no two movies produced with the same title.
  - or, there cannot be two distinct tuples in Movies table with equal values in title and year attribute.

- This is specified as follows.

```
CREATE TABLE Movies (
        title           VARCHAR(100),
        year            INT,
        length          INT,
        genre           CHAR(10),
        studioName      CHAR(30),
        producerC#      INT
        PRIMARY KEY     (title, year)        /* specifies primary   key */     );
```

A table may have multiple unique keys. Often it is ordered as per one unique key, called the primary key.

- or write

```
UNIQUE          (title, year)       /* specifies unique key */   );
```

Relational Algebra

- An algebra of operations over relations (or tables).
- Each operator acts on a relation or two relations and returns a relation (table).
- Gives simple and powerful ways to construct new relations from the given relations.
- Relational algebra expressions take as input stored tables (or database relations) and return answers to queries.
- SQL is a syntactic ``sugaring'' of relational algebra.
- DBMS internally first translates an SQL query into a relational algebra expression
  - or, a very similar expression tree notation: relational operator/operand tree notation.

Relational Algebra

- The algebra operates on relations (tables) whose schema is well-specified.
- Set operations on relations: Union , intersection and difference applied to relations.
- Let R and S be two relations (tables) with the exact same schema. This defines

Example

Table R with schema of MovieStar.

name	Address	gender	birthdate
Shahrukh Khan	1 Mannat, Mumbai 400001	M	1965-11-02
Amitabh Bachhan	Jalsa, Juhu Beach, Mumbai, 400049	M	1942-10-11

Table S with schema of MovieStar

name	Address	gender	birthdate
Amir Khan	Freeda Apt, Bandra, Mumbai 400001	M	1965-03-14
Shahrukh Khan	1 Mannat, Mumbai 400001	M	1965-11-02
Hema Malini	Shakunt, Juhu, Mumbai 400049	F	1948-10-16

$$R \cup S \; has \; 4 \; tuples \; (no \; duplicates)$$

name	Address	gender	birthdate
Shahrukh Khan	Mannat, Mumbai 400001	M	1965-11-02
Amitabh Bachhan	Jalsa, Juhu Beach, Mumbai, 400049	M	1942-10-11
Amir Khan	Freeda Apt, Bandra, Mumbai 400001	M	1965-03-14
Hema Malini	Shakunt, Juhu, Mumbai 400049	F	1948-10-16

Example R ∩ S, R - S

Table R with schema of Movies

name	Address	gender	birthdate
Shahrukh Khan	1 Mannat, Mumbai 400001	M	1965-11-02
Amitabh Bachhan	Jalsa, Juhu Beach, Mumbai, 400049	M	1942-10-11

Table S with schema of Movies

name	Address	gender	birthdate
Amir Khan	Freeda Apt, Bandra, Mumbai 400001	M	1965-03-14
Shahrukh Khan	1 Mannat, Mumbai 400001	M	1965-11-02
Hema Malini	Shakunt, Juhu, Mumbai 400049	F	1948-10-16

$$R \cap S \; has \; only \; 1 \; tuple$$

name	Address	gender	birthdate
Shahrukh Khan	1 Mannat, Mumbai 400001	M	1965-11-02

$$R - S \; also \; has \; only \; 1 \; tuple$$

name	Address	gender	birthdate
Amitabh Bachhan	Jalsa, Juhu Beach, Mumbai, 400049	M	1942-10-11

Selection Operator

- σ takes a logical predicate P and applies to an operand table R.

Table Movies

title	year	length	genre	studioName	producerC#
Sholay	1975	198	action	United Producer	1
Chak De! India	2007	153	sports	Yash Raj	3
3 Idiots	2009	171	comedy	Vinod Chopra Films	2

  - Find the value of the expression
  - This gives the tuples in the table Movies corresponding to movies whose length is at least 2hrs and 50 mins.

title	year	length	genre	studioName	producerC#
Sholay	1975	198	action	United Producer	1
3 Idiots	2009	171	comedy	Vinod Chopra Films	3

- Order of tuples is not significant since a relation is a set of tuples.

Projection Operator π

$\pi_{A_1, A_2, \ldots A_n}(R)$ is a relation whose schema has only the columns $A_1, A_2, \ldots, A_n$ of R.

Movies

title	year	length	genre	studioName	producerC#
Sholay	1975	198	action	United Producer	1
Chak De! India	2007	153	sports	Yash Raj	3
3 Idiots	2009	171	comedy	Vinod Chopra Films	2

- Find the value of the expression

$\pi_{title, length, year}(Movies) = ¿$

title	year	length
Sholay	1975	198
Chak De! India	2007	153
3 Idiots	2009	171

- Project Movies onto the attribute genre.

genre
action
sports
comedy

Cartesian Product

- The Cartesian product (or cross-product) of two relations (tables) R and S is the set of pairs that can be formed
  - by choosing the first element of the pair to be an element of R,
  - and the second element of the pair to be an element of S.
- Denoted as
- Schema of Cartesian product: all attributes of R and S are included.
  - all attributes A of R is included and denoted as R.A.
  - All attributes B of S are included and denoted as S.B.

Example of cartesian product R X S

Table R

Tuples	A	B
r_1	1	2
r_2	2	3

Table S

Tuples	B	C	D
s_1	2	4	3
s_2	3	1	5
s_3	4	2	6

R X S

Tuples	R.A	R.B	S.B	S.C	S.D
(r_1, s_1)	1	2	2	4	3
(r_1, s_2)	1	2	3	1	5
(r_1, s_3)	1	2	4	2	6
(r_2, s_1)	2	3	2	4	3
(r, s_2)	2	3	3	1	5
(r_2, s_3)	2	3	4	2	6

1. R has tuples {s1,s2}.
2. S ={s1,s2,s3}.
3. R X S has 2 X 3 = 6 tuples.
4. RXS = {t1,t2}X{s1,s2,s3.
5. in RXS, attributes A,B of R are renamed as R.A, R.B.
6. Same with S.

Natural Join

- Often in a database schema, there are two relations whose schema have attributes with identical names (and types etc.).
- For e.g., suppose we (re-)define the movies database schema with two relations whose schema is as follows.
  1. Movies (title, year, length, genre) and
  2. StarsIn (title, year, starName)
- Here, the common attribute names title and year have the same ***intent***.
- The **natural join** of Movies and StarsIn is denoted as
- Its schema is (title, year, length, genre, starName).
  - 'common' attributes appear only once in the schema (no renaming).
  - A tuple m from Movies and s from StarsIn pair successfully if
    - m and s agree on title and year attribute values.
    - i.e., the starName actor in tuple s worked in the movie tuple m.

Movies

title	year	length	genre
Sholay	1975	198	action
Chak De! India	2007	153	sports
3 Idiots	2009	171	comedy

Natural Join has no unmatched pair.
No dangling tuples!

StarsIn

title	year	name
Sholay	1975	Amitabh Bachhan
Sholay	1975	Hema Malini
Chak De! India	2007	Shahrukh Khan
Lagaan	2001	Amir Khan

$Movies \bowtie StarsIn$

title	year	length	genre	name
Sholay	1975	198	action	Amitabh Bachhan
Sholay	1975	198	action	Hema Malini
Chak De! India	2007	153	sports	Shahrukh Khan

- The 'Sholay' tuple matches with two actor names in StarsIn giving 2 tuples in the natural join.
- The 'Chak De! India' movie tuple matches with 1 actor name.
- Total 2+1 = 3 tuples in the natural join.
- '3 Idiots' in Movie and 'Lagaan' in StarsIn are each respectively unmatched.

**Theta Joins

Theta Joins is more general than natural joins.

- Natural joins equated all shared attributes.
  - Assumes that the shared attributes in both tuples refer to the same data items/entities.
- Theta-Joins allows pairing tuples from two relations on some other grounds.
  - Take the cartesian product of R and S.
  - Select from R X S only those tuples that satisfy condition P.

Example: Theta Join

R

A	B	C
1	2	3
6	7	8
9	7	8

S

B	C	D
2	3	4
2	3	5
7	8	10

$R \times S$

R.A	R.B	R.C	S.B	S.C	S.D
1	2	3	2	3	4
1	2	3	2	3	5
1	2	3	7	8	10
6	7	8	2	3	5
6	7	8	2	3	5
6	7	8	7	8	10
9	7	8	2	3	4
9	7	8	2	3	5
9	7	8	7	8	10

$R \bowtie_{R.A<S.D \text{ and } R.B \neq S.B} S$

R.A	R.B	R.C	S.B	S.C	S.D
1	2	3	7	8	10

Combining Operations to Form Queries

- Like any algebra, relational algebra allows us to form complex expressions
  - by applying operations to the result of other operations.
- Will represent relational algebra expressions also as
  - expressions and as expression trees (easy to visualize!)

- Query 1: From Movies relation, find all tuples of movies that have 'action' genre.

  Movies(title, year, length, genre, studioname, producer#)
- Soln:

- Expression tree:

$$\sigma_{genre = 'action'}$$

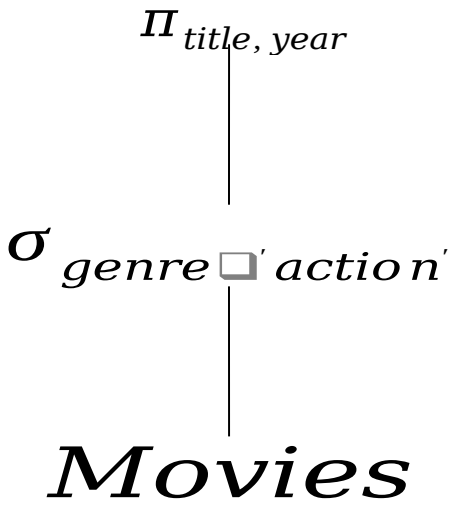
$$|$$

$$Movies$$

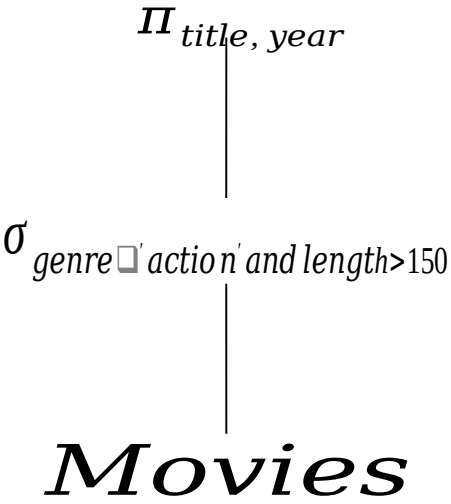
Combining Operators

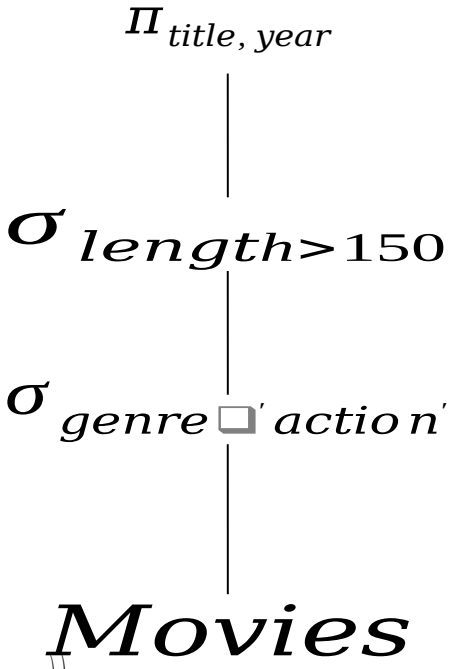
- Query 2: Find title and year (to unambiguously identify a movie) of movies that have the genre 'action'.
- Soln: we can take the result of query 1 and project on title, year.

$$\pi_{title,\ year}(\sigma_{genre='action'}(Movies))$$



- Query 3: Find title and year of movies that have the genre 'action' and length > 150.
- Soln: we can take the result of query 1 and project on title, year.
- Two Expression trees:





$$\pi_{title,\ year}(\sigma_{genre='action'\ and\ length>150}(Movies))$$

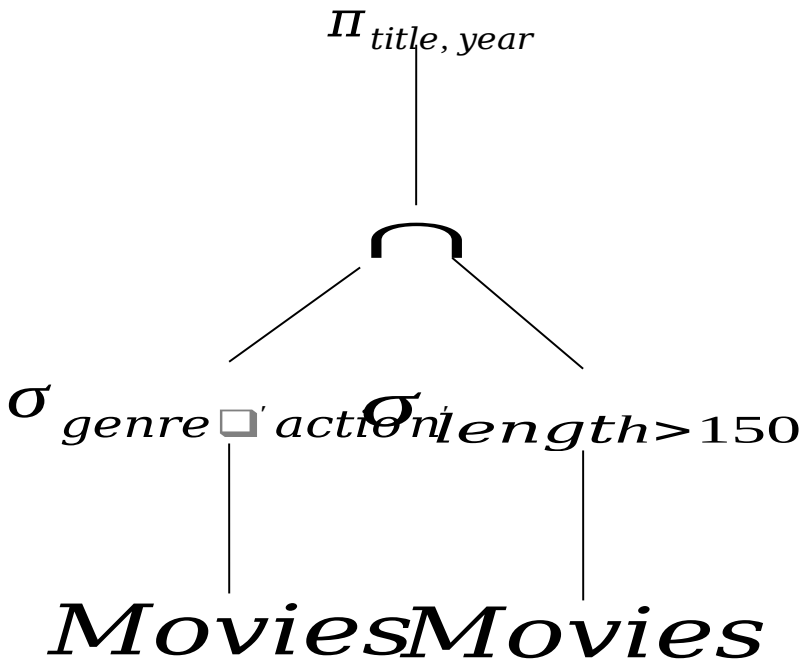
Relational Algebra

- Yet another expression for:

  Find title and year of movies that have the genre 'action' and length > 150.

  - *A query has many equivalent relational algebra expressions*

$$\pi_{title,\,year}\left(\sigma_{length>150}(Movies) \cap \sigma_{genre='action'}(Movies)\right)$$



Relational Query

- Schema:

  Movies(title, year, length, genre)

  StarsIn(movieTitle, movieYear, starName)

- *Query 1*: Find all genre='action' movies where 'Amitabh Bachhan' stars in.
  - Return title and year of the movie.
- Query 1 algebra:
  - Do a *theta join* of Movies with StarsIn and return title, year and starName.
    - Set starName='Amitabh Bachhan', set genre='action'.

  (

  ))

Relational Query

- As in algebra, we can give a name to a query expression and use it later.

  (

  ))

- *Query 2*: Find all genre = 'action' movies where 'Amitabh Bachhan' **does not** star in. Return title and year.
- Algebra steps:
  1. Find all title, year value pairs for movies of genre = 'action'.
  2. From this relation, remove (subtract) all title, year value pairs for movies of genre = 'action' of genre movies where 'Amitabh Bachhan' stars in.

Examples

➢ **Query1** : Find all years where 'Shahrukh Khan' starred in every action movie released in that year.

1. Find all title, year pairs of genre='action' that 'Shahrukh Khan' starred in.

S

)

2. Find all title, year pairs of genre='action'.

)

3. Find all title, year pairs of genre='action' that 'Shahrukh Khan' did not star in.

$W =$

4. Find all years where there was an action movie in which 'Shahrukh Khan' did not star in.

U =

5. Find all years where an action movie was released.

or)

6. Find all years where 'Shahrukh Khan' starred in every action movie released in that year.

Answer:

Query 1a: In all those years where 'Shahrukh Khan acted in all action movies released that year, also return the name and titles of all those action movies in each of those years.

- We extend our solution from the previous query.
  - = set of all years where Shahrukh Khan acted in each action movie released that year.
  - Z = set of all title, year pairs of all action movies released.
  - Natural join of W with Y. Schema is (title,year)

*Some exercises.

1. Find all (starName, year, genre) triples where a movie star with starName s starred in every genre g movie released in year y.

2. Find all (starName1, starName2, year, genre) tuples where are distinct star names such that in year y, starred in all movies released in that year with genre g but did not star in any of these movies.

   - Variation: Find tuples such that starred in all movies released in that year y with genre g but did not star in at least one of that genre g movie that year.

   - In the above questions, how does your answer stand with respect to 4 tuples where,
     1. there was no release of a movie of genre g movie in year .
     2. there might have been releases of genre g movie in year but actor did not star in any release of genre g movie in year

Can you count using Relational Algebra

Kids learn to count

- Given a relation R.
- Can we write these queries?
  - Does R have odd or even number of tuples?
  - How many tuples (records) does R have?
- NO!! Relational Algebra can't help us count !!
- What about this?
  - Give all the release years where Amitabh Bachhan acted in at least five movies released that year.
  - YES !! A bit complicated- Theta join over 5 copies of movies table.
- What about?
  - Give the year in which Shahrukh Khan had the largest number of movies released compared to any other actor that year.
  - NO!! Once again Relational Algebra cannot count!
- We need some operators like COUNT, SUM, MIN, MAX etc.

Naming and Renaming Operator

- An operator for naming or renaming relations or attributes of relations.
- E.g., consider our Movies-StarsIn schema

  StarsIn(movieTitle, movieYear, starName)

  Movies(title, year, length, genre)
- Query:

Find all (title1, title2, year) triples where Amir Khan acted in two distinct action genre movies with titles title1 and title2 movies both released in the same year.

- Compute a relation AAMovies(title, year), which are all the action genre movies that Amir Khan acted in.

$$AAMovies \quad \Pi_{title,\,year}$$

$$\sigma_{movieTitle=title\ and\ movieYear=year\ and\ starName\square'Amir\ Khan'\ and\ genre\square'action'}$$

$$StarsIn \times Movies$$

1. Rename AAMovies(title, year) with name and schema as AAMov1(title1, year). (conceptually it is a copy with different names and attributes.)
2. Rename AAMovies(title, year) again this time with name and schema AAMov1(title2, year). /* conceptual second copy */

➢

Set of all triples (title1, title2, year) of action genre movies starring Amir Khan released the same year.

- Expressed using the renaming operator twice:

)

End (Lecture 2, 3)

Relational Algebra Examples

- Operator (R).
- It renames the attributes of the operand relation R.
- E.g., consider example schema
  - R(A,B,C) and S(B,C,D)
  - Then, R X S has schema (R.A, R.B, R.C, S.B, S.C, S.D).
  - Suppose we rename the attributes

  (R.A, R.B, R.C, S.B, S.C, S.D) as (E,F,G,H, I,J) .
  - Written as

    .
  - Equivalently, this is same as

    (S).