
Recap of database background

February 6, 2003

Based on

Chapter 3.1, 3.4, 3.6, 5.2-5.4, 6.1-6.5 in GUW

Advanced Database Technology, Spring 2003

— What we expect from you —

- Basic course in databases, e.g.,
 - Relational data model/ Relational databases
 - Relational algebra
 - SQL
- Basic course in algorithms and data structures, e.g.,
 - Search trees
 - Sorting algorithms
 - Hashing
 - Big-O notation
 - Basic algorithm analysis

— This part of the lecture

- Relational data model
 - Basic concepts, like attribute, schemas etc.
 - Functional dependency and keys
 - BCNF
- Relational algebra and SQL examples
 - Define relational algebra
 - Basic operations, like set operations, joins, selection etc.
 - Bags (multisets)
 - More operations, e.g., duplicate removal, grouping

More repetition may appear in later lectures when needed.

— The relational data model —

- The model behind most implementations of databases.
- Data is always represented as relations (2-dim. tables).
- SQL is a query language for relational databases and is based on relational algebra.

Some basic concepts:

- Relation
- Attribute
- Schema
- Tuple
- Key

How many of these do you recognize?

How many of these are you able to define now?

Relation

The way to represent data is through relations.

A **relation** is a two-dimensional table.

The order of rows and columns can be exchanged, and it is still the same relation.

Example:

<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>
Star Wars	1977	124	color
Mighty Ducks	1991	104	color
Wayne's World	1992	95	color

Attribute

An **attribute** is the name of a column in a relation. It usually describes the meaning of the content in the column.

Example:

<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>
Star Wars	1977	124	color
Mighty Ducks	1991	104	color
Wayne's World	1992	95	color

Schema

A **schema** is a description of a class of relation. It consists of the name of the relation and the set of attributes in the relation.

That it is a **set** of attributes means that the attributes are unordered.

Example:

<u>Movies</u>			
<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>
Star Wars	1977	124	color
Mighty Ducks	1991	104	color
Wayne's World	1992	95	color

Schema for the above relation:

Movies(title, year, length, filmType)

A set of schemas for relations are called a **design** or a **database schema**.

— Tuple —

A **tuple** is a row in a table. The values in the row are called components.

A relation can be seen as a set of tuples.

Example:

<u>Movies</u>			
<i>title</i>	<i>year</i>	<i>length</i>	<i>film Type</i>
Star Wars	1977	124	color
Mighty Ducks	1991	104	color
Wayne's World	1992	95	color

How many tuples are there? Name them.

— Functional dependency

Functional dependency is a central concept to discuss keys and normal forms.

Functional dependency:

If two or more tuples agree on attributes A_1, A_2, \dots, A_n , then they also agree on attribute B .

We say that:

A_1, A_2, \dots, A_n functionally determines B .

Written as:

$A_1 A_2 \cdots A_n \rightarrow B$

— Functional dependency —

Example:

<u>Movies</u>					
<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>	<i>studioName</i>	<i>starName</i>
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	1977	124	color	Fox	Harrison Ford
Mighty Ducks	1991	104	color	Disney	Emilio Estevez
Wayne's World	1992	95	color	Paramount	Dana Carvey
Wayne's World	1992	95	color	Paramount	Mike Meyers

A possible functional dependency:

title year → *length*

Not a possible functional dependency:

title year filmType → *starName*

Keys

A set of attributes $\{A_1, \dots, A_n\}$ is a **key** for a relation R if:

1. $\{A_1, \dots, A_n\}$ functionally determine all other attributes of R .
2. No $X \subset \{A_1, \dots, A_n\}$ functionally determines all other attributes of R .

If only 1 holds, then $\{A_1, \dots, A_n\}$ is called a **superkey**.

Example:

<u>Movies</u>					
<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>	<i>studioName</i>	<i>starName</i>
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	1977	124	color	Fox	Harrison Ford
Mighty Ducks	1991	104	color	Disney	Emilio Estevez

$\{title, year, starName\}$ is a key for Movies.

$\{title, year, length, starName\}$ is a superkey for Movies.

— Boyce-Codd Normal Form (BCNF) —

Bad design of schemas may lead to problems/anomalies:

- Redundancy.
- Update anomalies.
- Delete anomalies.

Avoid anomalies by defining schemas in Boyce-Codd Normal Form.

Example:

<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>	<i>studioName</i>	<i>starName</i>
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	1977	124	color	Fox	Harrison Ford
Mighty Ducks	1991	104	color	Disney	Emilio Estevez

— Boyce-Codd Normal Form (BCNF) —

A relation R is in **BCNF** if and only if:

Whenever there is a nontrivial functional dependency

$$A_1 A_2 \cdots A_n \rightarrow B \text{ for } R,$$

then $\{A_1, A_2, \dots, A_n\}$ is a superkey for R .

Example:

Decompose the relation to be in BCNF and thereby remove anomalies.

<u>Movies</u>				<u>Stars in</u>		
<i>title</i>	<i>year</i>	<i>length</i>	<i>studioN.</i>	<i>title</i>	<i>year</i>	<i>starN.</i>
Star Wars	1977	124	Fox	Star Wars	1977	Carrie F.
Mighty D.	1991	104	Disney	Star Wars	1977	Mark H.
Wayne's W.	1992	95	Param.	Star Wars	1977	Harrison F.
				Mighty D.	1991	Emilio E.
				Wayne's W.	1992	Dana C.
				Wayne's W.	1992	Mike M.

— Relational algebra and SQL examples —

Relational algebra is notation for operations on relations, like constructing new relations and defining queries on relations.

The rest of the lecture is about:

- Basic operations in **relational algebra**:
 - **set operations** (e.g. union)
 - **selection** and **projection**
 - **join**
- **Bags** (multisets). Why they are used and what the consequence is.
- More operations, e.g., duplicate removal, grouping

Illustrated with a few examples in SQL.

How many of the above red words do you recognize?

How many of them are you able to define now?

— Set operations —

Operations on two sets R and S , where R and S must have the same set of attributes. We have the three set operations:

- Union, $R \cup S$,
- Intersection, $R \cap S$, and
- Difference, $R \setminus S$.

Example:

<u>Movies1</u>			<u>Movies2</u>		
<i>title</i>	<i>year</i>	<i>filmType</i>	<i>title</i>	<i>year</i>	<i>filmType</i>
Star Wars	1977	color	Star Wars	1999	color
Mighty Ducks	1991	color	Mighty Ducks	1991	color
Wayne's World	1992	color	Star Wars	2002	color

— Projection

A **projection** of relation R on attributes A_1, \dots, A_n is denoted by

$$\pi_{A_1, \dots, A_n}(R)$$

and is the relation R restricted to columns for attributes A_1, \dots, A_n .

<i>title</i>	<i>year</i>	<i>length</i>	<i>filmType</i>	<i>studioName</i>	<i>starName</i>
Star Wars	1977	124	color	Fox	Carrie Fisher
Star Wars	1977	124	color	Fox	Mark Hamill
Star Wars	1977	124	color	Fox	Harrison Ford
Mighty Ducks	1991	104	color	Disney	Emilio Estevez

	<i>title</i>	<i>length</i>	<i>studioName</i>
$\pi_{title, length, studioName}(Movies) =$	Star Wars	124	Fox
	Mighty Ducks	104	Disney

Selection

A **selection** of tuples satisfying condition C from relation R is denoted by

$$\sigma_C(R)$$

and is the relation R restricted to tuples for which condition C is satisfied.

C can be any boolean expression, i.e. it may involve multiple attributes, constants, AND, OR, and NOT.

Example:

Movies		
<i>title</i>	<i>year</i>	<i>filmType</i>
Star Wars	1977	color
Mighty Ducks	1991	color
Wayne's World	1992	color

$$\sigma_{year > 1981}(Movies) =$$

<i>title</i>	<i>year</i>	<i>filmType</i>
Mighty Ducks	1991	color
Wayne's World	1992	color

— Join (natural)

Natural-Join or Inner-Join:

Let R and S be two relations with attributes R_1, \dots, R_n and S_1, \dots, S_m respectively. The join of relations R and S , denoted

$$R \bowtie S$$

has attributes $\{R_1, \dots, R_n\} \cup \{S_1, \dots, S_m\}$.

If $r \in R$ and $s \in S$ agree on attributes $\{R_1, \dots, R_n\} \cap \{S_1, \dots, S_m\}$ then the joint tuple for r and s is in $R \bowtie S$.

There are other types of join, e.g., Theta-Join and Outer-Join.

Join example

<u>Movies</u>				<u>Stars in</u>		
<i>title</i>	<i>year</i>	<i>length</i>	<i>studioN.</i>	<i>title</i>	<i>year</i>	<i>starN.</i>
Star Wars	1977	124	Fox	Star Wars	1977	Carrie F.
Mighty D.	1991	104	Disney	Star Wars	1977	Mark H.
Wayne's W.	1992	95	Param.	Star Wars	1977	Harrison F.
				Mighty D.	1991	Emilio E.
				Wayne's W.	1992	Dana C.
				Wayne's W.	1992	Mike M.

Movies ⋈ Stars in =

<i>title</i>	<i>year</i>	<i>length</i>	<i>studioN.</i>	<i>starN.</i>
Star Wars	1977	124	Fox	Carrie F.
Star Wars	1977	124	Fox	Mark H.
Star Wars	1977	124	Fox	Harrison F.
Mighty D.	1991	104	Disney	Emilio E.
Wayne's W.	1992	95	Param.	Dana C.
Wayne's W.	1992	95	Param.	Mike M.

Bags

Relational algebra is an algebra on sets, but most database systems do not only use sets, they also use bags.

A **bag** or a multiset, is a set where elements may appear more than once. (In a relation there may be two or more identical rows.)

The motivation for using bags instead of sets is that some operations can be implemented faster. E.g.,

- union
- projection

— Operations on bags vs. sets —

Some examples of the difference between operations on bags and sets.

- $R \cup S$: All rows in R and S , even if they appear in both or if they appear more than once in R or in S .
- $R \cap S$: if tuple t appears n times in R and m times in S , then it appears $\min(n, m)$ times in $R \cap S$.
- $\pi_{A_1, \dots, A_n}(R)$ (projection): All tuples in R also appear in $\pi_{A_1, \dots, A_n}(R)$, even if the rows become identical when some columns are removed.

<u>Movies1</u>			<u>Movies2</u>		
<i>title</i>	<i>year</i>	<i>filmType</i>	<i>title</i>	<i>year</i>	<i>filmType</i>
Star Wars	1977	color	Star Wars	1999	color
Mighty Ducks	1991	color	Mighty Ducks	1991	color
Wayne's World	1992	color	Star Wars	2002	color

— More operations —

Other useful relational operations often used in languages like SQL:

- Duplicate elimination: When bags are used it is useful to be able to get rid of duplicates.
- Aggregation operators: E.g., sum, average, maximum in a column.
- Grouping: Divide a relation up into groups of tuples depending on the values in one or more attributes.
- Extended projection: Creation of new columns from existing columns by performing some kind of computation.

SQL

SQL is a language that can be used for expressing queries on relations. It is based on a “mixture” of relational algebra for sets and bags.

Some SQL examples:

- `SELECT A_1, \dots, A_n FROM R` means $\pi_{A_1, \dots, A_n}(R)$.
- `SELECT * FROM R WHERE C` means $\sigma_C(R)$.
- `R UNION S` means $R \cup S$.
- `R EXCEPT S` means $R \setminus S$.
- `R NATURAL JOIN S` means $R \bowtie S$.
- `SELECT DISTINCT * FROM R` means $\delta(R)$.

— SQL update operations —

SQL also supports the creation and modification of relations.

Some SQL examples:

- `CREATE TABLE R (<schema description>)`
- `INSERT INTO R VALUES (v_1, \dots, v_n).`
- `DELETE FROM R WHERE C .`
- `UPDATE R SET $A = v$ WHERE C .`

Finally, SQL can be used to specify what indexes should be created, e.g.:

- `CREATE INDEX I ON $R(A)$`

Summary

This part of the lecture was about:

- the relational data model
- relational algebra
- relational algebra on bags
- some examples in SQL

These concepts will underlie much of the first part of the course.

See title-slide for recommended reading.