Database

Université Grenoble Alpes

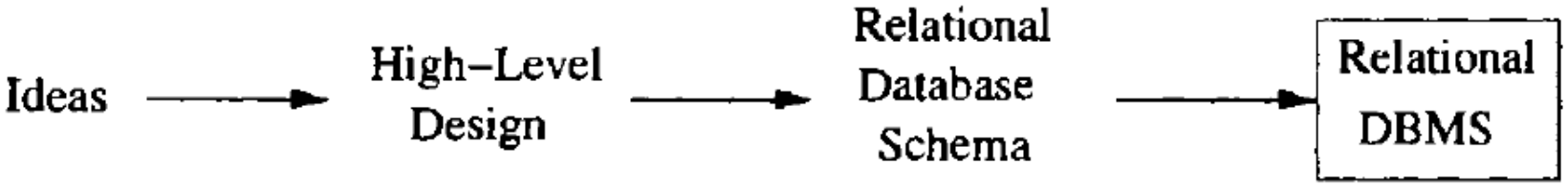
02.03.2023

bahareh.afshinpour@univ-grenoble-alpes.fr

Main reference:
A First Course in Database Systems (and associated material) by
J. Ullman and J. Widom, Prentice-Hall

Recall

- In practice, it is often easier to start with a higher-level model and then convert the design to the relational model.

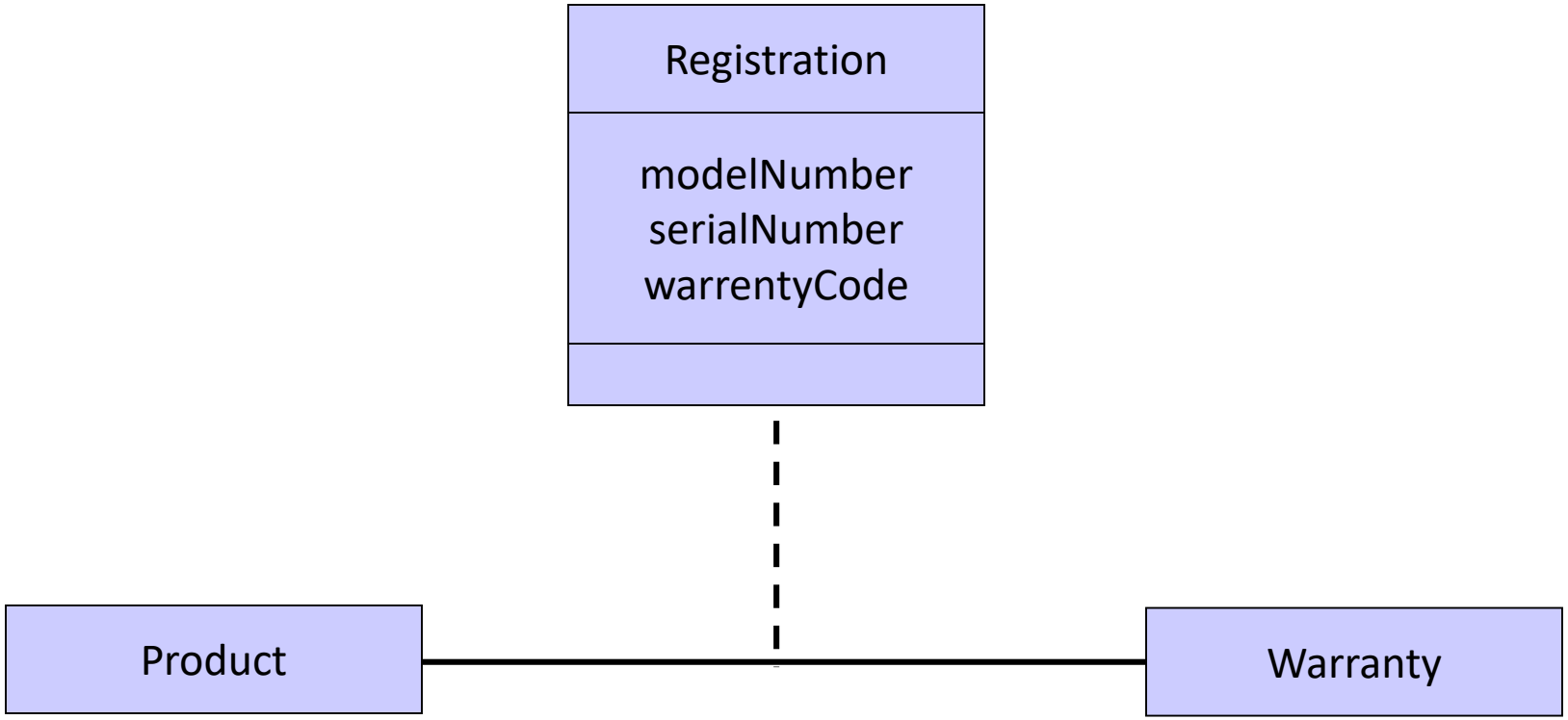


The database modeling and implementation process

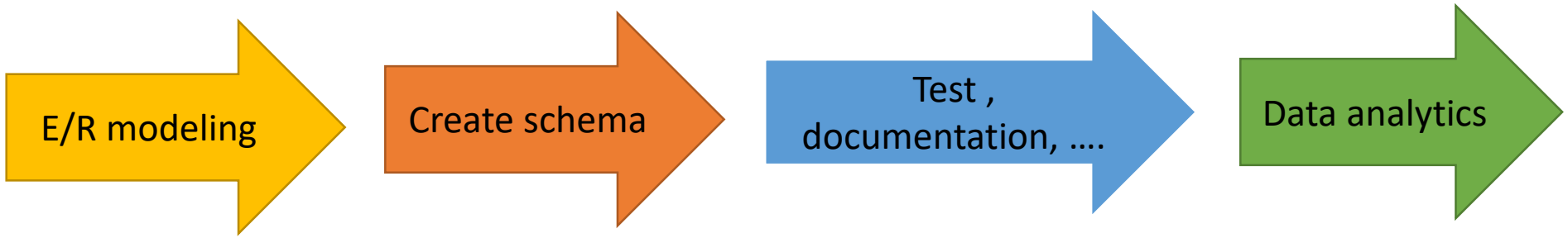
- There are several options for the notation in which the design is expressed.
  - Entity-relationship diagram
  - UML (class diagram)
  - ODL(object description language)

Association Relationships

Associations can also be objects themselves, called *link classes* or an *association classes*.



Top view



-ERD is much more top-level.
-Type of the attributes will be shown in Schema
-In the actual implementation of the database, you need to think about the testing, Supporting it, monitoring it, …
-Data analytics is compared to machine learning in data science. Analytics is gathering data, analysis is interpreting data. you can extract key elements from the data so that you can learn about your customers, your products.

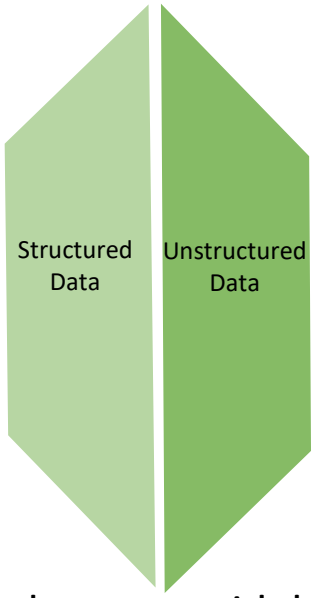
Top view

Relational database

SQL server

Oracle

MySQL



Not only SQL DBs

Cassandra

Couchbase

MongoDB

- I just wanted to focus my attention on, the most widely used ones.
- There are two distinct categories here,
  - there is the structured data,
  - and unstructured data.
- you don't always need a relational database. Sometimes you need a database that is designed for the purposes of collecting, gathering, and processing real-time data.

NoSQL Databases

- A NoSQL referring to non-SQL or non-relational is a database that provides a mechanism for the storage and retrieval of data.
- A NoSQL database includes simplicity of design and finer control over availability. The data structures used by NoSQL databases are different from those used by default in relational databases which makes some operations faster in NoSQL.
- Advantages of NoSQL

  There are many advantages of working with NoSQL databases such as MongoDB and Cassandra. The main advantages are high scalability and high availability.
- **Disadvantages of NoSQL**
  - NoSQL is an open-source database.
  - GUI is not available
  - Backup is a weak point for some NoSQL databases like MongoDB.

https://www.geeksforgeeks.org/types-of-databases/

Algebric and Logical query Languages

Chapter 5

Programming

- We now switch our attention from modeling to programming for relational databases.
- We have two abstract programming language
  - Relational algebra(chapter 2)
  - Logic-based

- We extend the algebra so it can handle several more operations than were described previously.

Relational Algebra Operations

Chapter 2 presented the classical relational algebra.

- The usual set operations: union, intersection, difference

- Operations that remove parts of relations:
    - selection, projection
- Operations that combine tuples from two relations:
    - Cartesian product, join
- Since each operation returns a relation,
    - operations can be *composed*!

Projection

- The Projection operator applied to a relation R, produces a new relation with a subset of R's columns.

title	*year*	*length*	*genre*	*studioName*	*producerC#*
Star Wars	1977	124	sciFi	Fox	12345
Galaxy Quest	1999	104	comedy	DreamWorks	67890
Wayne's World	1992	95	comedy	Paramount	99999

The relation Movies

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

title	*year*	*length*
Star Wars	1977	124
Galaxy Quest	1999	104
Wayne's World	1992	95

$\pi_{genre}(\text{Movies})$

genre
sciFi
comedy

Duplicate tuples are eliminated

Selection

- The selection operator applied to a relation R, produces a new relation with a **subset of R's tuples**.
- The tuples in the resulting relation are those that satisfy some condition C that involves the attributes R.

title	*year*	*length*	*genre*	*studioName*	*producerC#*
Star Wars	1977	124	sciFi	Fox	12345
Galaxy Quest	1999	104	comedy	DreamWorks	67890
Wayne's World	1992	95	comedy	Paramount	99999

The relation Movies

$$\sigma_{length \geq 100}(\text{Movies})$$

title	*year*	*length*	*inColor*	*studioName*	*producerC#*
Star Wars	1977	124	sciFi	Fox	12345
Galaxy Quest	1999	104	comedy	DreamWorks	67890

Cartesian Product (cross-product)

- The Cartesian Product of two sets R and S is the set of pairs that can be formed by choosing the first element from R and the second from S.
- If R and S have some attributes in common, we need to invent a new name for the identical attributes.

Relation R

A	B
1	2
3	4

Relation S

B	C	D
2	5	6
4	7	8
9	10	11

Relation R X S

A	R.B	S.B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

Bags

In this section, we shall consider relations that are bags (multisets) rather than sets. That is, we shall allow the same tuple to appear more than once in a relation. When relations are bags, there are changes that need to be made to the definition of some relational operations, as we shall see.

5.1.1 Why Bags?

As we mentioned, commercial DBMS's implement relations that are bags, rather than sets. An important motivation for relations as bags is that some relational operations are considerably more efficient if we use the bag model. For example:

1. To take the union of two relations as bags, we simply copy one relation and add to the copy all the tuples of the other relation. There is no need to eliminate duplicate copies of a tuple that happens to be in both relations.

2. When we project relation as sets, we need to compare each projected tuple with all the other projected tuples, to make sure that each projection appears only once. However, if we can accept a bag as the result, then we simply project each tuple and add it to the result; no comparison with other projected tuples is necessary.

Chapter 5 introduced the modifications necessary to treat relations as **bags** of tuples rather that sets.

Union, Intersection, and Difference

- In the bag union $R \cup S$, tuple t appears $n + m$ times.
- In the bag intersection $R \cap S$, tuple t appears $\min(n, m)$ times.
- In the bag difference $R - S$, tuple t appears $\max(0, n - m)$ times. That is, if tuple t appears in R more times than it appears in S, then t appears in $R - S$ the number of times it appears in R, minus the number of times it appears in S. However, if t appears at least as many times in S as it appears in R, then t does not appear at all in $R - S$. Intuitively, occurrences of t in S each "cancel" one occurrence in R.

Example

Example 5.4: Let R be the relation of Fig. 5.1, that is, a bag in which tuple $(1,2)$ appears three times and $(3,4)$ appears once. Let S be the bag

A	B
1	2
3	4
3	4
5	6

R:
(1,2) 3times
(3,4) 1 time

Then the bag union $R \cup S$ is the bag in which $(1,2)$ appears four times (three times for its occurrences in R and once for its occurrence in S); $(3,4)$ appears three times, and $(5,6)$ appears once.

The bag intersection $R \cap S$ is the bag

A	B
1	2
3	4

with one occurrence each of $(1,2)$ and $(3,4)$. That is, $(1,2)$ appears three times in R and once in S, and $\min(3,1) = 1$, so $(1,2)$ appears once in $R \cap S$. Similarly, $(3,4)$ appears $\min(1,2) = 1$ time in $R \cap S$. Tuple $(5,6)$, which appears once in S but zero times in R appears $\min(0,1) = 0$ times in $R \cap S$. In this case, the result happens to be a set, but any set is also a bag.

The bag difference $R - S$ is the bag

A	B
1	2
1	2

To see why, notice that $(1, 2)$ appears three times in R and once in S, so in $R - S$ it appears $\max(0, 3 - 1) = 2$ times. Tuple $(3, 4)$ appears once in R and twice in S, so in $R - S$ it appears $\max(0, 1 - 2) = 0$ times. No other tuple appears in R, so there can be no other tuples in $R - S$.

Selection on Bags

- To apply a selection to a bag, we apply the selection condition to each tuple independently. As always with bags, we do not eliminate duplicate tuples in the result.

Example 5.5: If R is the bag

A	B	C
1	2	5
3	4	6
1	2	7
1	2	7

then the result of the bag-selection $\sigma_{C \geq 6}(R)$ is

A	B	C
3	4	6
1	2	7
1	2	7

Products of Bags

The rule for the Cartesian product of bags is the expected one. Each tuple of one relation is paired with each tuple of the other, regardless of whether it is a duplicate or not. As a result, if a tuple r appears in a relation R m times, and tuple s appears n times in relation S, then in the product $R \times S$, the tuple rs will appear mn times.

Extended Operators of Relational Algebra

SQL have several other operations that have proved quite important in applications.

1-The duplicate-elimination operators

2-Aggregation operators such as SUM . They apply to attributes(columns) of a relation.

3-Grouping of tuples, can partition the tuples into group. Aggregation can then be applied to columns within each group.

4- Extended projection gives additional power to the operation projection.

5-The sorting operator turns a relation into a list of tuples, sorted according to one or more attributes.

6-The outerjoin operator is a variant of the join that avoids dangling tuples.

Examples

- **lives**(person-name,street,city)
- **works**(person-name, company-name,salary)
- **located-in**(company-name,city)
- **manages**(person-name,manager-name)

For the above schema (the primary key for each relation is denoted by the underlined attribute), provide relational algebra expressions for the following queries:

1. Find all tuples in works of all persons who work for the City Bank company (which is a specific company in the database).
   (a) $\sigma_{(cname='City\ Bank')}(works)$

2. Find the name of persons working at City Bank who earn more than $50,000.
   (a) $\pi_{pname}(\sigma_{(cname='City\ Bank')\wedge(salary>50000)}(works))$

3. Find the name and city of all persons who work for City Bank and earn more than 50,000.
Similar to previous query, except we have to access the lives table to extract the city of the employee . Note the join condition in the query.
   (a) $\pi_{lives.pname,lives.city}(\sigma_{((cname='City\ Bank')\wedge(salary>50000)\wedge(lives.pname=works.pname))})(lives\times works)$

Duplicate Elimination $\delta(R)$

- Sometimes we need an operator that converts a bag to a set.

Example 5.8: If R is the relation

A	B
1	2
3	4
1	2
1	2

from Fig. 5.1, then $\delta(R)$ is

A	B
1	2
3	4

Aggregation operators (summarize)

- Many operators we can apply to set or bags of numbers or strings.
- They are used to **summarize or aggregate** the values in one column of relation

- For example:
- SUM
- AVG
- MIN and MAX (numerical values and character-string values)
- COUNT

Example

Example 5.9: Consider the relation

A	B
1	2
3	4
1	2
1	2

Some examples of aggregations on the attributes of this relation are:

1. `SUM(B)` $= 2 + 4 + 2 + 2 = 10$.
2. `AVG(A)` $= (1 + 3 + 1 + 1)/4 = 1.5$.
3. `MIN(A)` $= 1$.
4. `MAX(B)` $= 4$.
5. `COUNT(A)` $= 4$.

□

The Grouping Operator $\gamma_L(R)$

- We introduce an operator that allows us to group a relation and aggregate some columns.
- γL(R)
- A list L of elements, each of which is either:
    1) **A grouping attribute**: an attribute of the relation R to which the γ is applied.
    2) **An aggregated attribute**: an attribute of the relation R to which an aggregate operator is applied.

The Grouping Operator

The relation returned by the expression $\gamma_L(R)$ is constructed as follows:

1. Partition the tuples of R into *groups*. Each group consists of all tuples having one particular assignment of values to the grouping attributes in the list L. If there are no grouping attributes, the entire relation R is one group.

2. For each group, produce one tuple consisting of:

   i. The grouping attributes' values for that group and

   ii. The aggregations, over all tuples of that group, for the aggregated attributes on list L.

Example

R =

A	B	C
1	2	3
4	5	6
1	2	5

$\gamma_{A,B,AVG(C)}$ (R) = ??

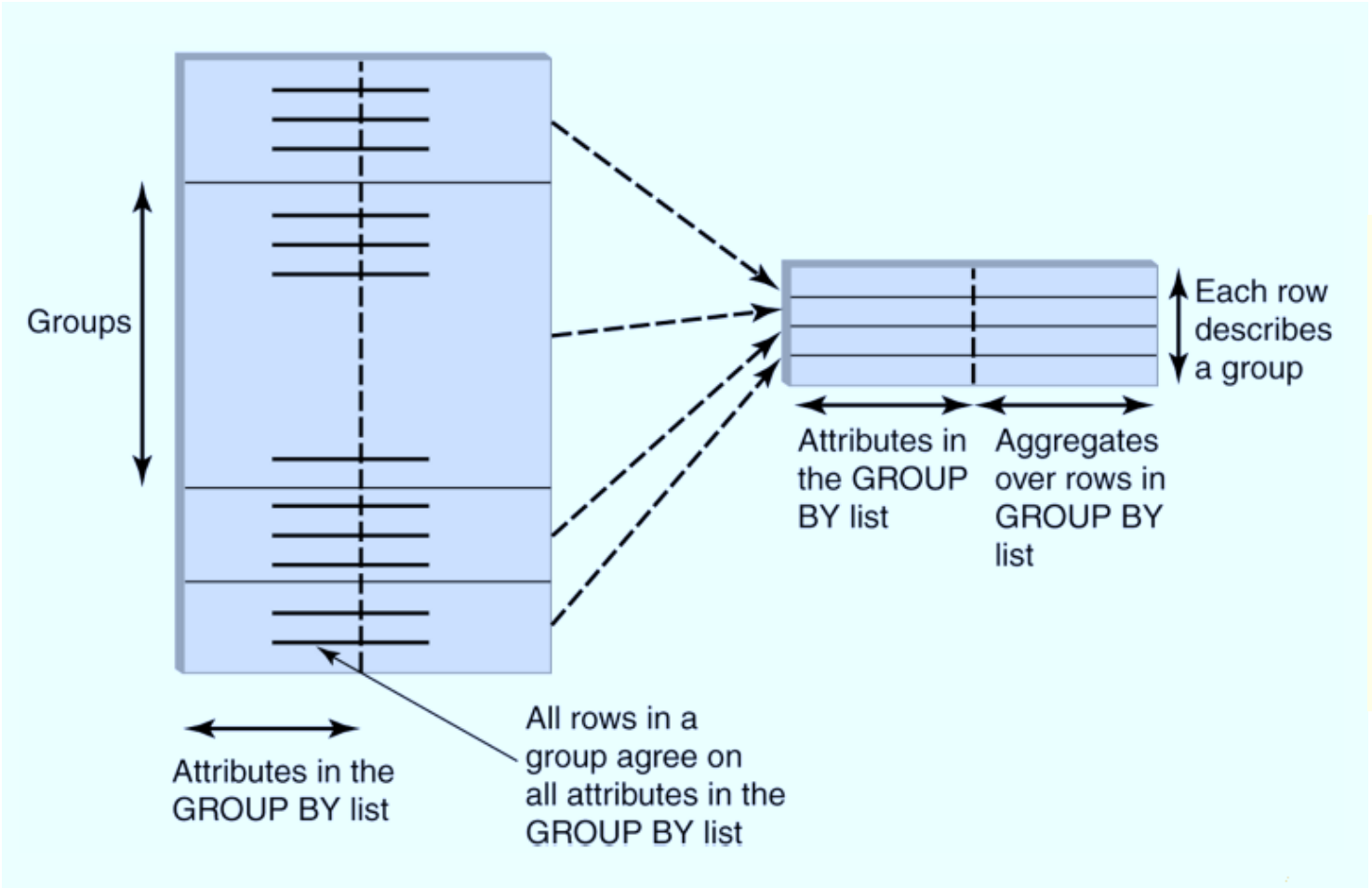
Result has grouping attributes and aggregations as attributes.
There is one tuple for each list of values for the grouping attributes and their group's aggregations.

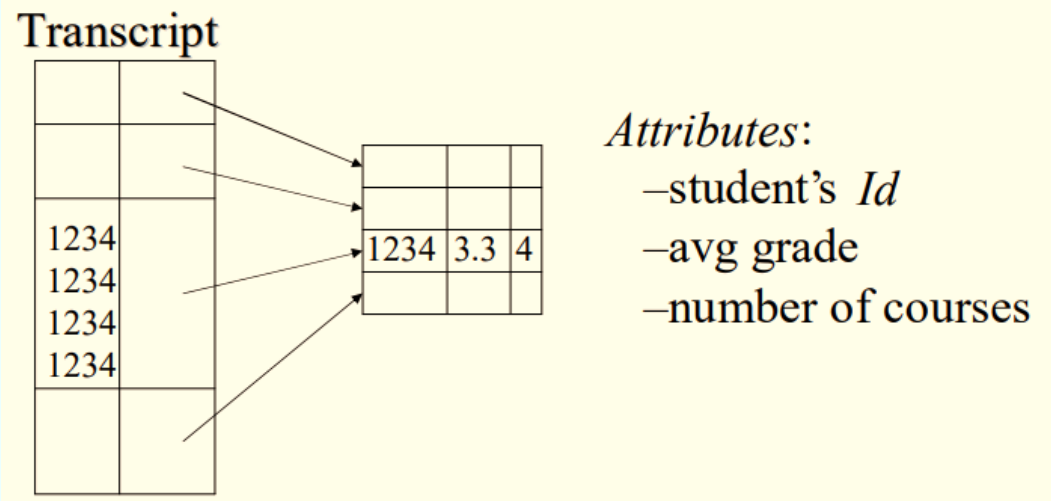
First, group *R* :

A	B	C
1	2	3
1	2	5
4	5	6

Then, average *C* within groups:

A	B	AVG(C)
1	2	4
4	5	6


Groups
Each row describes a group
Attributes in the GROUP BY list
Aggregates over rows in GROUP BY list
Attributes in the GROUP BY list
All rows in a group agree on all attributes in the GROUP BY list

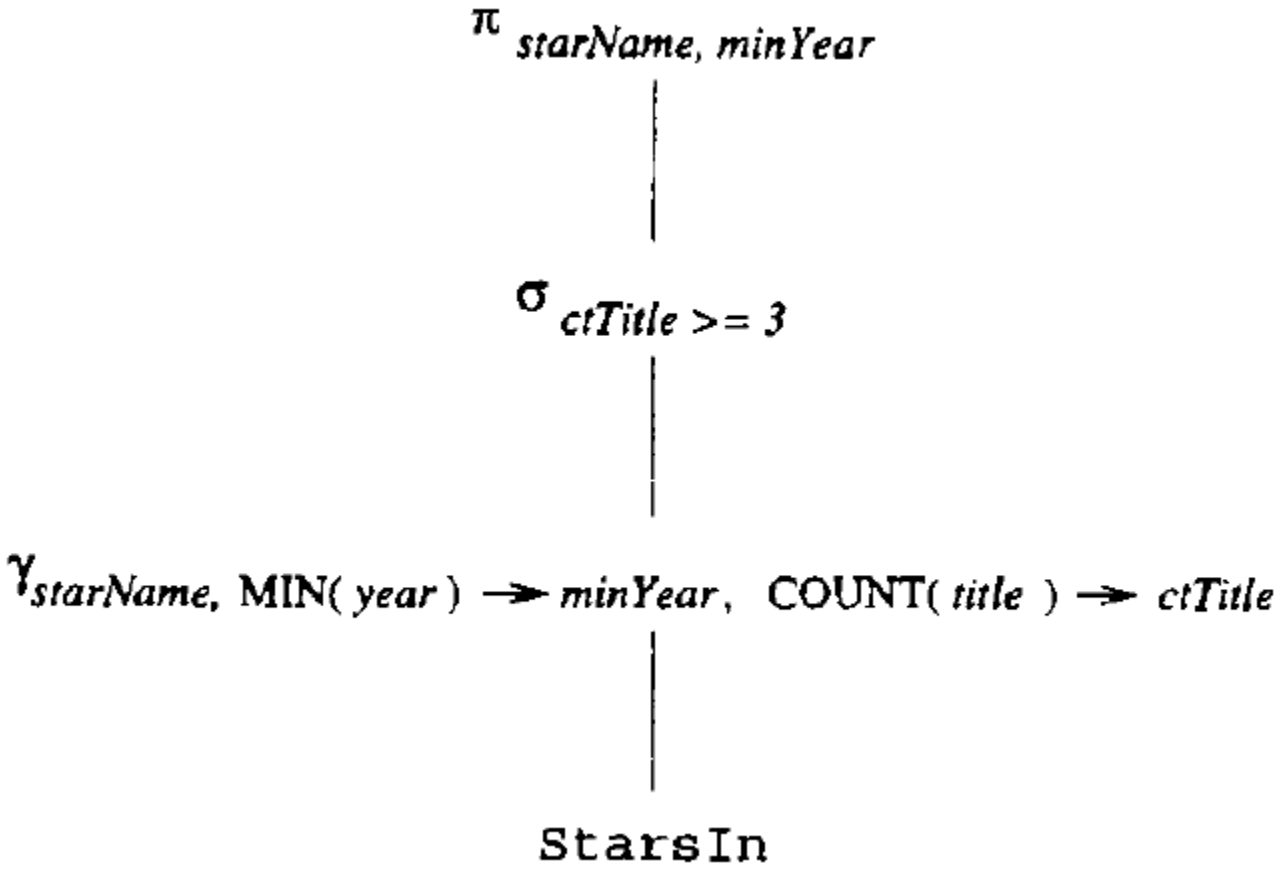

Transcript
1234
1234
1234
1234
1234 3.3 4
Attributes:
–student's *Id*
–avg grade
–number of courses

Example: Grouping/Aggregation

- **StarsIn(**title, year, starName**)**
- Suppose we want, for each star who has appeared in at least three movies the earliest year in which he appeared.

  Movies, starIn, stars

  - First we group, using starName as a grouping attribute.
  - Then, we have to compute the MIN(year) for each group.
  - However, we need also compute COUNT(title) aggregate for each group, in order to filter out those stars with less than three movies.



for each star who has appeared in at least three movies the earliest year in which he appeared.

$\sigma_{ctTitle>3}[\gamma_{starName,MIN(year)\rightarrow minYear,COUNT(title)\rightarrow ctTitle}(StarsIn)]$

Extending the projection Operator

An expression $x \to y$, where x and y are names for attributes. The element $x \to y$ in the list L asks that we take the attribute x of R and *rename* it y; i.e., the name of this attribute in the schema of the result relation is y.

An expression $E \to z$, where E is an expression involving attributes of R, constants, arithmetic operators, and string operators, an
name for the attribute that results from the calculation impli
example, $a+b \to x$ as a list element represents the sum of the
and b, renamed x. Element $c \,||\, d \to e$ means concatenate the
string-valued attributes c and d and call the result e.

Example 5.11 : Let R be the relation

A	B	C
0	1	2
0	1	2
3	4	5

Then the result of $\pi_{A,B+C \to X}(R)$ is

A	X
0	3
0	3
3	9

The sorting Operator

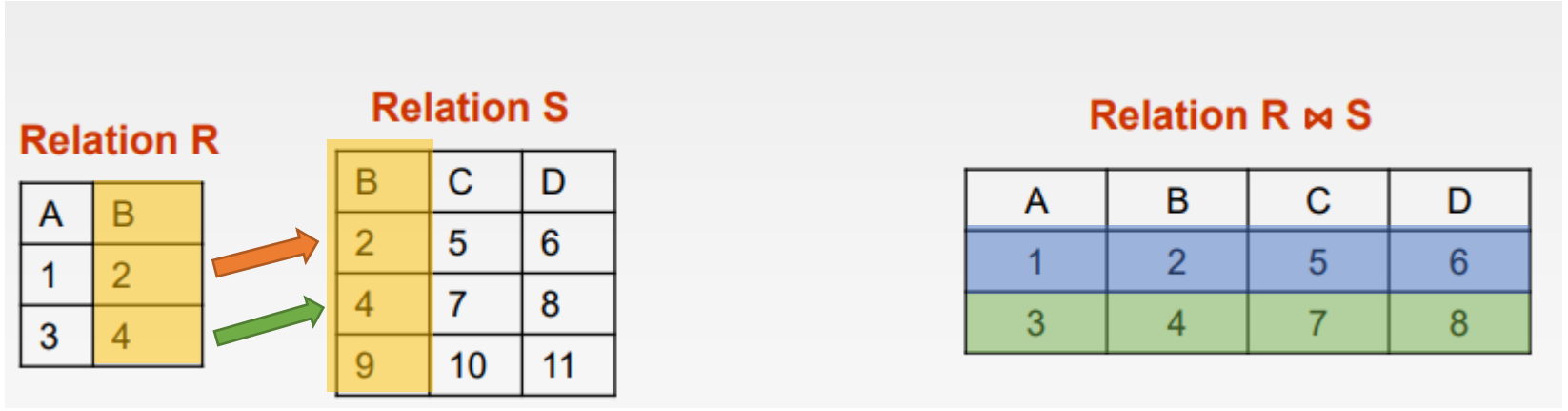
- More efficient
- More easily find tuple

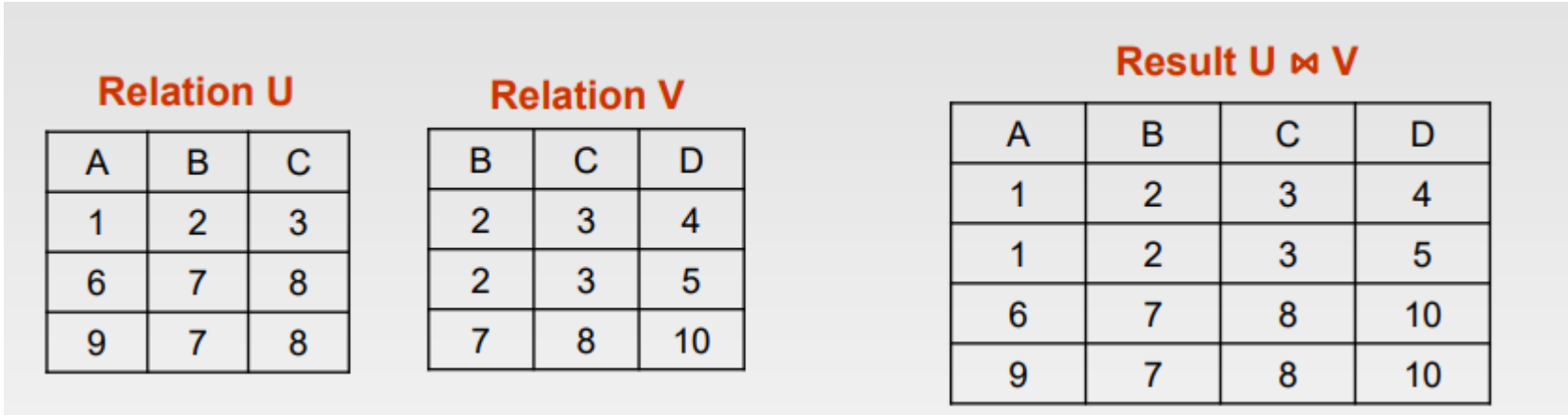
The expression $\tau_L(R)$, where R is a relation and L a list of some of R's attributes, is the relation R, but with the tuples of R sorted in the order indicated by L. If L is the list $A_1, A_2, \ldots, A_n$, then the tuples of R are sorted first by their value of attribute A_1. Ties are broken according to the value of A_2; tuples that agree on both A_1 and A_2 are ordered according to their value of A_3, and so on. Ties that remain after attribute A_n is considered may be ordered arbitrarily.

Example 5.12: If R is a relation with schema $R(A, B, C)$, then $\tau_{C,B}(R)$ orders the tuples of R by their value of C, and tuples with the same C-value are ordered by their B value. Tuples that agree on both B and C may be ordered arbitrarily. □

Natural Joins

- The Natural join of two sets R and S is the set of pairs that agree in whatever attributes are common to the schemas of R and S.





ITEM_ID	ITEM_NAME	ITEM_UNIT	COMPANY_ID
1	Chex Mix	Pcs	16
6	Cheez-It	Pcs	15
2	BN Biscuit	Pcs	15
3	Mighty Munch	Pcs	17
4	Pot Rice	Pcs	15
5	Jaffa Cakes	Pcs	18
7	Salt n Shake	Pcs	-

COMPANY_ID	COMPANY_NAME	COMPANY_CITY
18	Order All	Boston
15	Jack Hill Ltd	London
16	Akas Foods	Delhi
17	Foodies.	London
19	sip-n-Bite.	New York

** Same column came once

COMPANY_ID	ITEM_ID	ITEM_NAME	ITEM_UNIT	COMPANY_NAME	COMPANY_CITY
16	1	Chex Mix	Pcs	Akas Foods	Delhi
15	6	Cheez-It	Pcs	Jack Hill Ltd	London
15	2	BN Biscuit	Pcs	Jack Hill Ltd	London
17	3	Mighty Munch	Pcs	Foodies.	London
15	4	Pot Rice	Pcs	Jack Hill Ltd	London
18	5	Jaffa Cakes	Pcs	Order All	Boston

$$\textit{Notation: } R \bowtie S$$

Purpose: relate rows from two tables, and

- enforce equality on all common attributes
- eliminate one copy of common attributes

Outer join

- Fill in missing fields with nulls

We shall consider the "natural" case first, where the join is on equated values of all attributes in common to the two relations. The *outerjoin* $R \overset{\circ}{\bowtie} S$ is formed by starting with $R \bowtie S$, and adding any dangling tuples from R or S. The added tuples must be padded with a special *null* symbol, $\perp$, in all the attributes that they do not possess but that appear in the join result. Note that $\perp$ is written `NULL` in SQL (recall Section 2.3.4).

A	B	C
1	2	3
4	5	6
7	8	9

(a) Relation U

B	C	D
2	3	10
2	3	11
6	7	12

(b) Relation V

A	B	C	D
1	2	3	10
1	2	3	11
4	5	6	$\perp$
7	8	9	$\perp$
$\perp$	6	7	12

(c) Result $U \overset{\circ}{\bowtie} V$

Left Outerjoin Example

For an example consider the tables *Employee* and *Dept* and their left outer join:

Employee

Name	EmpID	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

DeptName	Mgr
Sales	Harriet

Name	EmpID	DeptName	Mgr
Harry	3415	Finance	null
Sally	2241	Sales	Harriet
George	3401	Finance	null
Harriet	2202	Sales	Harriet

Example

Exercise 5.2.1: Here are two relations:

$R(A, B)$: $\{(0, 1),\ (2, 3),\ (0, 1),\ (2, 4),\ (3, 4)\}$

$S(B, C)$: $\{(0, 1),\ (2, 4),\ (2, 5),\ (3, 4),\ (0, 2),\ (3, 4)\}$

Compute the following: a) $\pi_{A+B,A^2,B^2}(R)$; b) $\pi_{B+1,C-1}(S)$; c) $\tau_{B,A}(R)$; d) $\tau_{B,C}(S)$; e) $\delta(R)$; f) $\delta(S)$; g) $\gamma_{A,\,\mathrm{SUM}(B)}(R)$; h) $\gamma_{B,\mathrm{AVG}(C)}(S)$; ! i) $\gamma_A(R)$; ! j) $\gamma_{A,\mathrm{MAX}(C)}(R \bowtie S)$; k) $R \overset{\circ}{\bowtie}_L S$; l) $R \overset{\circ}{\bowtie}_R S$; m) $R \overset{\circ}{\bowtie} S$; n) $R \overset{\circ}{\bowtie}_{R.B<S.B} S$.

Left outer join- Right outer join

There are many variants of the basic (natural) outerjoin idea. The *left outerjoin* $R \overset{\circ}{\bowtie}_L S$ is like the outerjoin, but only dangling tuples of the left argument R are padded with $\perp$ and added to the result. The *right outerjoin* $R \overset{\circ}{\bowtie}_R S$ is like the outerjoin, but only the dangling tuples of the right argument S are padded with $\perp$ and added to the result.

Oracle

- Oracle has been the de facto premier database that has excelled worldwide in every area of the production database environment.

- Oracle:
  - how to navigate the graphical interface
  - SQL basics (Structured Query Language)
  - making it proficient with Oracle environment

- Oracle databases are usually considered for environments that are working with medium to large datasets. Example: test messages

Oracle:Multimodel database

- ORACLE supports the document model with supports JSON, XML and text-based data natively within the database.
- ORACLE also supports graphing data, which is designed for organic association. It is perfect for detecting fraud.
- ORACLE also supports what is referred to as Triple Store, and that gives developers the ability to add meaning to graph data.
- Oracle is the best database out there, it's been around for a very long time. And as a result, Oracle has had time to learn from its mistakes.

Oracle Storage Architecture

- These are the six files that are utilized while the database is running.

parameter file

- a parameter file. It is used to configure the database instance at the time that it's running. Specifically, memory, along with other resources that the user utilizes.

password file

- This file allows users with specific roles to connect remotely so that they can perform admin tasks.

backup file

- This is used to recover the database in the event that, it happens to crash.

archived redo log files

- These are the ones that contain the **ongoing history of changes** made to the database.

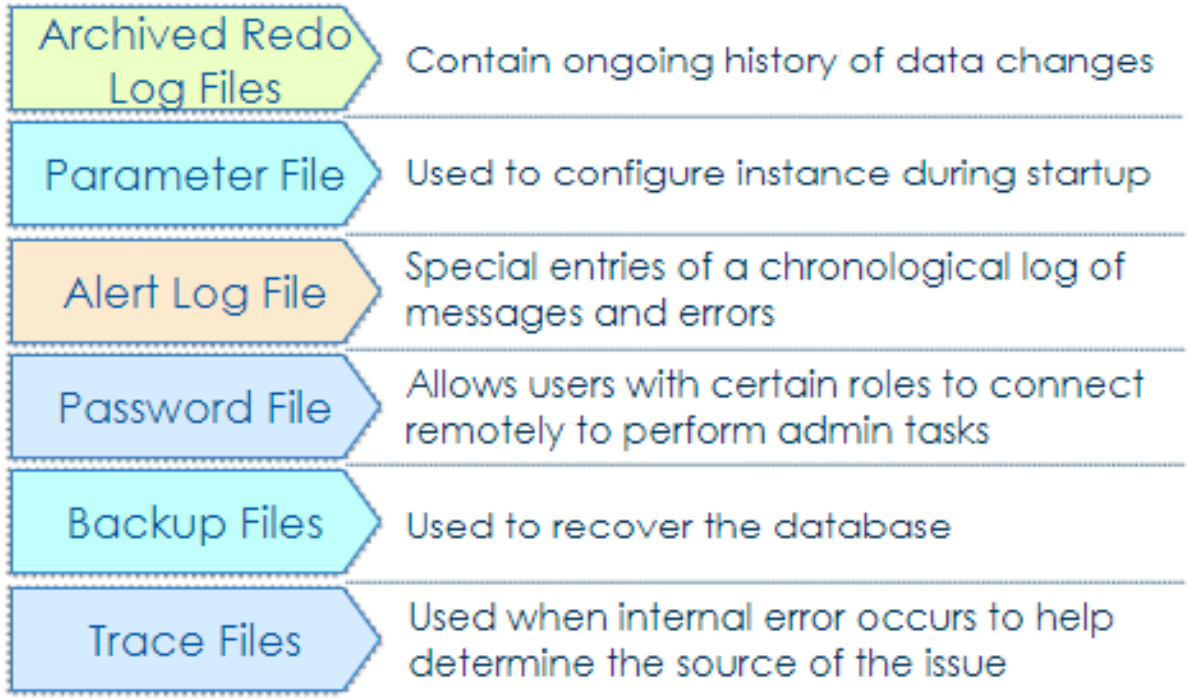
trace files

- These are used when **internal errors** occur to help determine the source of the issue. After all, without trace files, it would be very difficult to figure out why some problems happened.

alert log files

- These are special entries of a chronological log of messages and errors. If you wanted to take a look at what happened even historically, when some problems occurred or **how the system was functioning**, you could keep track of these alert logs based on dates to see what's going on in the system.

Oracle Storage Architecture



Install Oracle

- you can download your copy of Oracle Database 21c for Windows 10 x64 version for FREE.
- This download is going to be around 3GB in size.
- Currently Oracle Database 21c is only available for
  - Linux x86-64,
  - HP-UX ia64 and
  - Microsoft Windows x64 (64bit) operating system.
- Unfortunately Oracle 21c is not available for 32 bit Operating systems.