CSCI317 Database Performance Tuning

Decompositions

Dr Janusz R. Getta

School of Computing and Information Technology -
University of Wollongong

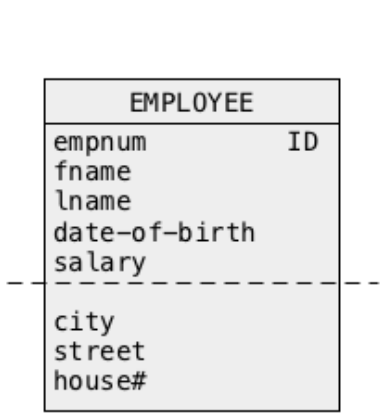
Decompositions

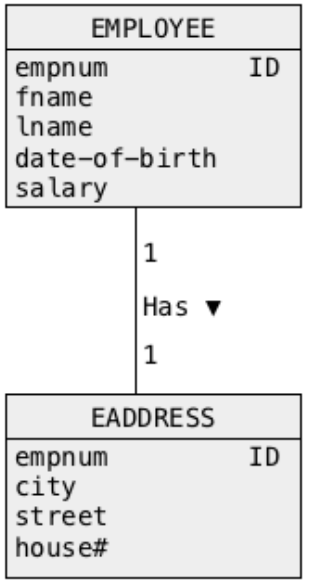
Outline

Horizontal decomposition

An objective of horizontal decomposition of a class of objects (vertical decomposition of a relational table) is to reduce a size of objects in a class (rows in a relational table)

For example, we would like to reduce the size of objects in a class `EMPLOYEE`





Horizontal decomposition

Performance related observations

Horizontal decomposition reduces the amounts of persistent storage needed to implement decomposed classes of objects (relational tables)

Reduction of object size (row size) speeds up processing of sequential scans of classes of objects (relational tables) because less data blocks are need to store data

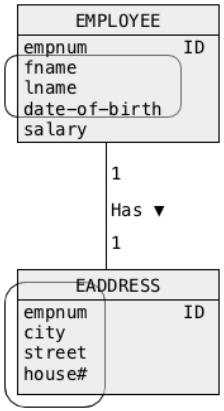
Horizontal decomposition may increase processing time when the values of attributes from disjoint classes of objects (relational tables) are used in the same query; then additional join is required

Horizontal decomposition does not affect normalization

After horizontal decomposition insertion of a new object (row) needs two operations and not one like before

Horizontal decomposition

Sample applications



Queries that benefit from horizontal decomposition

- Find full address (city,street,house#) of an employee identified by a given employee number
- Find the first and the last name of all employees more than 60 years old

Queries that loose on horizontal decomposition

- Find full address (city,street,house#) of all employees more than 60 years old
- Find the first and the last name of all employees living in a given city

Decompositions

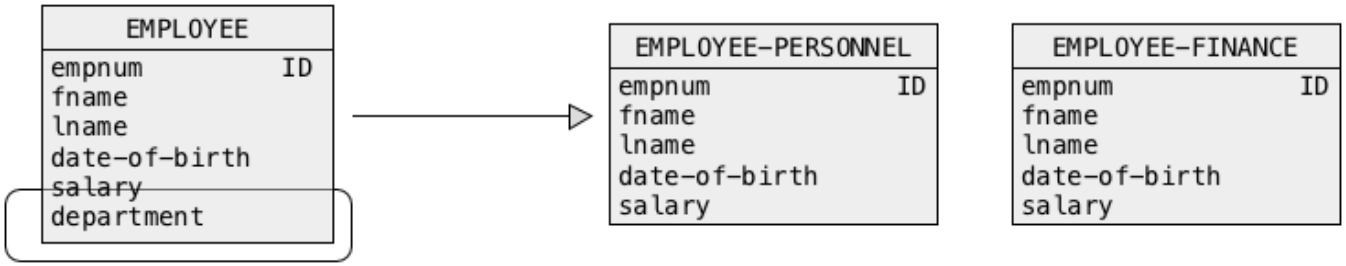
Outline

- Horizontal decomposition
- Vertical decomposition
- Implementations

Vertical decomposition

An objective of vertical decomposition of a class of objects (horizontal decomposition of a relational table) is to reduce the total number of objects in the classes (rows in the relational tables) obtained after decomposition

For example, we would like to reduce the total number of objects in a class `EMPLOYEE` that contains information about employees from finance and personnel departments



Vertical decomposition

Performance related observations

Vertical decomposition reduces the total number of objects accessed by an application

Reduction of total number of objects (total number of rows) speeds up processing of sequential scans of classes of objects (relational tables)

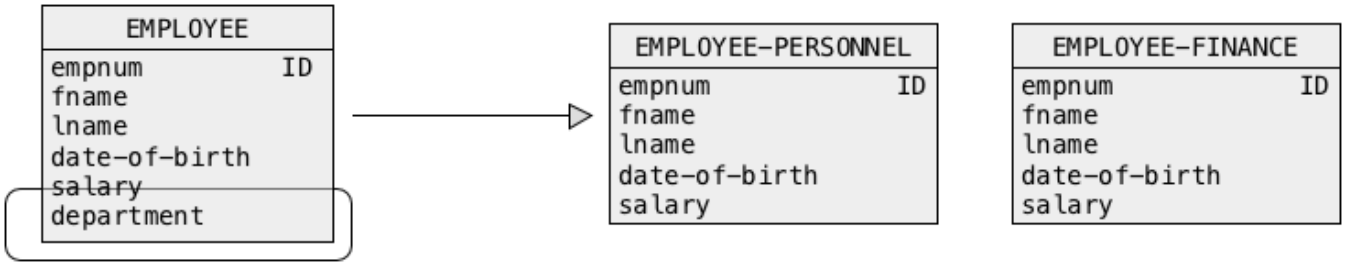
Vertical decomposition increases processing time when all objects from the decomposed classes of objects (relational tables) must be accessed; then additional union is required

Vertical decomposition does not affect normalization

Vertical decomposition requires replication of code that enforces consistency constraints, e.g. stored procedures/functions or triggers

Vertical decomposition

Sample applications



Queries that benefit from vertical decomposition

- Find a salary of a given employee from finance department
- Find the names of employees from personnel department

Queries that loose on vertical decomposition

- Find an average salary of employees from both finance and personnel departments
- Find the total number of employees in the given age ranges

Decompositions

Outline

Implementations

Vertical decomposition of relational tables

- Vertical decomposition can be implemented either through indexing or materialization
- A composite key index can be created on the attributes included in a relational table obtained from horizontal decomposition
- A materialized view can be created to implement a relational table obtained from vertical decomposition

Horizontal decomposition of relational tables

- Horizontal decomposition can be implemented partitioned relational tables
- Larger Database Management Systems implement partitioned relational tables
- A concept of partitioned relational table is a component of standard SQL