Uni Freiburg, Web Science Group
Prof. Peter Fischer
Systems Infrastructure for Data Science - Winter 2012/13

Exercise Sheet #7: Horizontal and Vertical Fragmentation

December 14, 2012

Exercise 7.1 : Horizontal Fragmentation

Consider the following two relation samples from a customer database of a large enterprise. (Note that the category of customers is determined based on whether their balance is greater than the minimum balance for that category.)

`Promotions(category, min_balance, percent_discount)`

`Customers(id, name, email, balance, category)`

`Promotions`

category	min_balance	percent_discount
basic	0	0
silver	25000	5
gold	50000	10
platinum	100000	15

`Customers`

id	name	email	balance	category
1	J. Doe	doe@gmail.com	12000	basic
2	M. Smith	smith@gmail.com	122000	platinum
3	A. Lee	lee@gmail.com	65000	gold
4	J. Miller	miller@gmail.com	8000	basic
5	B. Casey	casey@gmail.com	100	basic
6	L. Chu	chu@gmail.com	30000	silver
7	R. Davis	davis@gmail.com	26000	silver
8	J. Jones	jones@gmail.com	90000	gold

A. Show the owner-member relationship between these two relations.

B. Assume the following two applications that access these relations:

* Determine the discount percentage for different customer categories based on their minimum balances.
* Find the names and emails of customers of a given customer category in order to send them emails about current discounts.

Perform horizontal fragmentation (primary and derived) on these two relations.

C. Show the correctness of your fragmentation.

Exercise 7.2 : Vertical Fragmentation

There is a database that looks as follows:

`Employees (id, firstname, lastname, address, picture, family_status, toe_length)`
`Salaries (id, employee_id, month, amount, note, printed_pdf)`

There is also a unique key constraint on Salaries(employee_id, month).

To fullfill their purpose, your business processes need to execute the following queries:

- Compute the total amount of salaries payed to each employee:
  `select employee_id, sum(amount) from salaries group by employee_id;`
- Show all salaries payed during a given month along with the name of the employee:
  `select emp.firstname, emp.lastname, sal.amount from employees emp, salaries sal where sal.employee_id = emp.id and month='2009-11';`

A. Come up with a useful vertical fragmentation that helps to answer the above queries faster.

B. Draw the attribute usage matrix and the attribute affinity matrix for the scenarion above. To compute the attribute affinity matrix, assume `query access` equals 1 in all cases.

C. Stretch your mind! Would it make sense to vertically fragment a table as far as one key-value pair per attribute? E.g. the Salaries table above would be fragmented into S1(id,employee_id), S2(id,month), S3(id,amount), S4(id,note) and S5(id,printed_pdf).