TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Dr. Thomas Neumann

Exercises for *Foundations in Data Engineering*, WiSe 23/24

Alexander Beischl, Maximilian Reif (i3fde@in.tum.de)
http://db.in.tum.de/teaching/ws2324/foundationsde

Sheet Nr. 12

Exercise 1 Regarding the CAP theorem, it was mentioned in the lecture, that only two of the three "wishes" (consistency, availability, partition tolerance) can be met simultaneously.

However, which of the three combinations CA, CP, AP are very similar?

Exercise 2 Which guarantees that ACID databases provide can noSQL databases give up to achieve higher performance? For every trick: How does it work and why is it faster?

Exercise 3 For the following cases: For which database guarantees would it make sense to give them up to gain performance without jeopardizing the usefulness of the whole process?

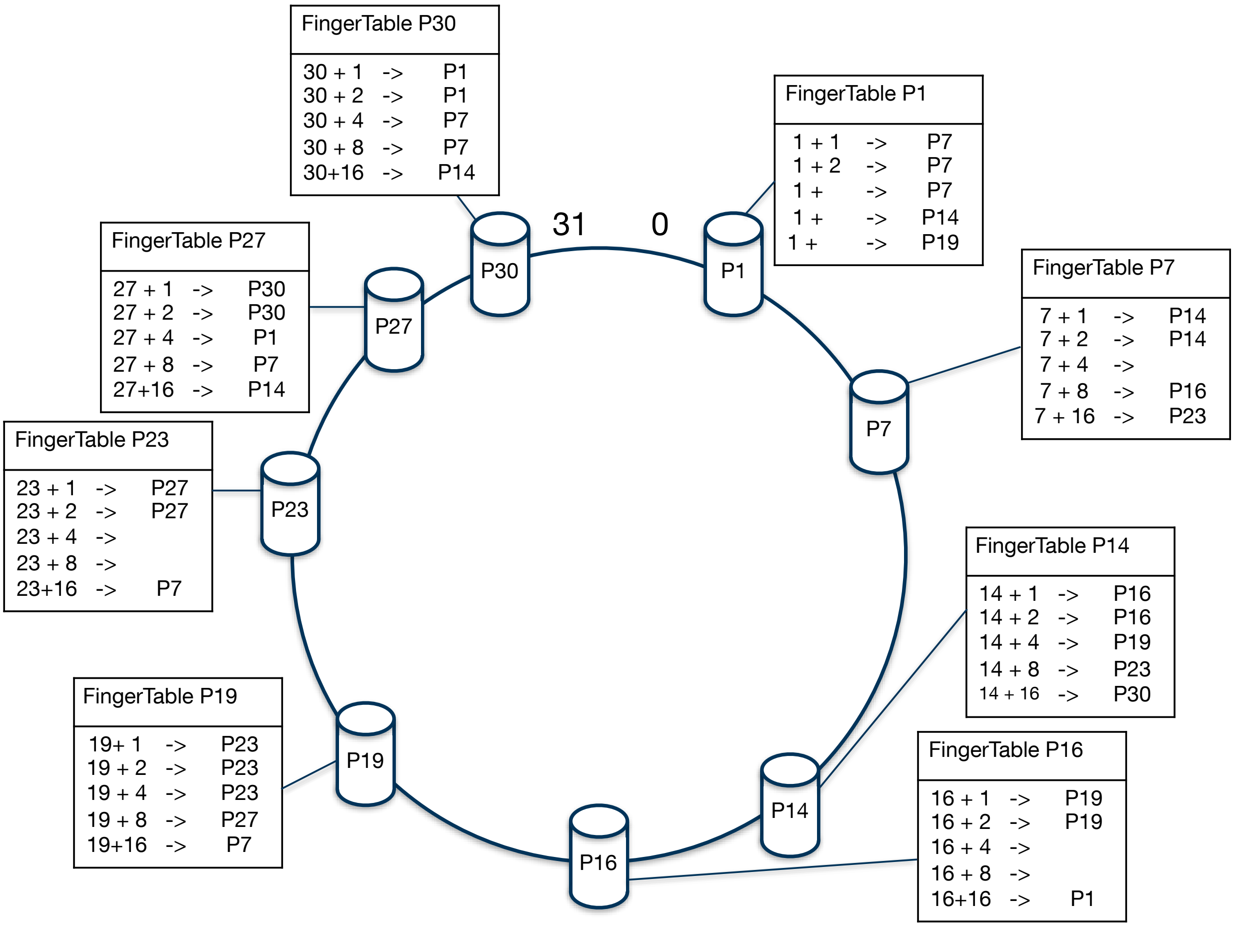
1. All cars of manufacturer X send traffic information to X's servers. X uses it to predict traffic jams.
2. All cars of X have an emergency call function to connect to the company's head quarter and also send location information.
3. Telecom provider Y saves all call statistics in a database in order to bill clients.
4. University U collects weather information from weather stations around the country.
5. Small company C stores lists of parts in a database.

Exercise 4 Imagine a document database that is distributed over many computers. Every document has a key that uniquely identifies it. Whenever a new document is added to the database, this is done using a key and the document content. Documents can be retrieved with the key.

1. Can you find real-world examples of such a database?
2. The documents in the system should be distributed over n participating computers so that roughly the same number of documents reside on every machine. Furthermore, it should be quite simple to determine from a given key on which machine the corresponding document resides, e.g. using hashing. Describe a strategy to accomplish this.
3. How can the above strategy be modified so that computers can be added to the system while documents are still distributed evenly?
4. Describe a strategy in this system to find the document that corresponds to a given key.
5. How can this strategy be changed, so that only very few documents need to be moved when a machines joins the system?
6. Can you devise a strategy in which no node is a single point of failure, but that does not need to send network request to all participating machines?

Exercise 5 Figure 1 shows a Chord network with 8 participating peers (compute nodes).

1. Complete the missing entries in the finger tables.
2. Search key 24 starting at peer 30.
3. Show that the chord search scheme always results in a maximum of $log(n)$ steps, where n is the ring size. (Hint: Have a look at the steps needed to find key 24.)



FingerTable P30		
30 + 1	->	P1
30 + 2	->	P1
30 + 4	->	P7
30 + 8	->	P7
30+16	->	P14

FingerTable P1		
1 + 1	->	P7
1 + 2	->	P7
1 +	->	P7
1 +	->	P14
1 +	->	P19

FingerTable P27		
27 + 1	->	P30
27 + 2	->	P30
27 + 4	->	P1
27 + 8	->	P7
27+16	->	P14

FingerTable P7		
7 + 1	->	P14
7 + 2	->	P14
7 + 4	->	
7 + 8	->	P16
7 + 16	->	P23

FingerTable P23		
23 + 1	->	P27
23 + 2	->	P27
23 + 4	->	
23 + 8	->	
23+16	->	P7

FingerTable P14		
14 + 1	->	P16
14 + 2	->	P16
14 + 4	->	P19
14 + 8	->	P23
14 + 16	->	P30

FingerTable P19		
19+ 1	->	P23
19 + 2	->	P23
19 + 4	->	P23
19 + 8	->	P27
19+16	->	P7

FingerTable P16		
16 + 1	->	P19
16 + 2	->	P19
16 + 4	->	
16 + 8	->	
16+16	->	P1

Figure 1: Example of a Chord overlay-network.