Memory:
The New Disk

TU Berlin, 2010-11-12

Who?

- Tim Lossen / **@tlossen**
- Ruby developer
- backend developer at ....

wooga – world of gaming

www.wooga.com

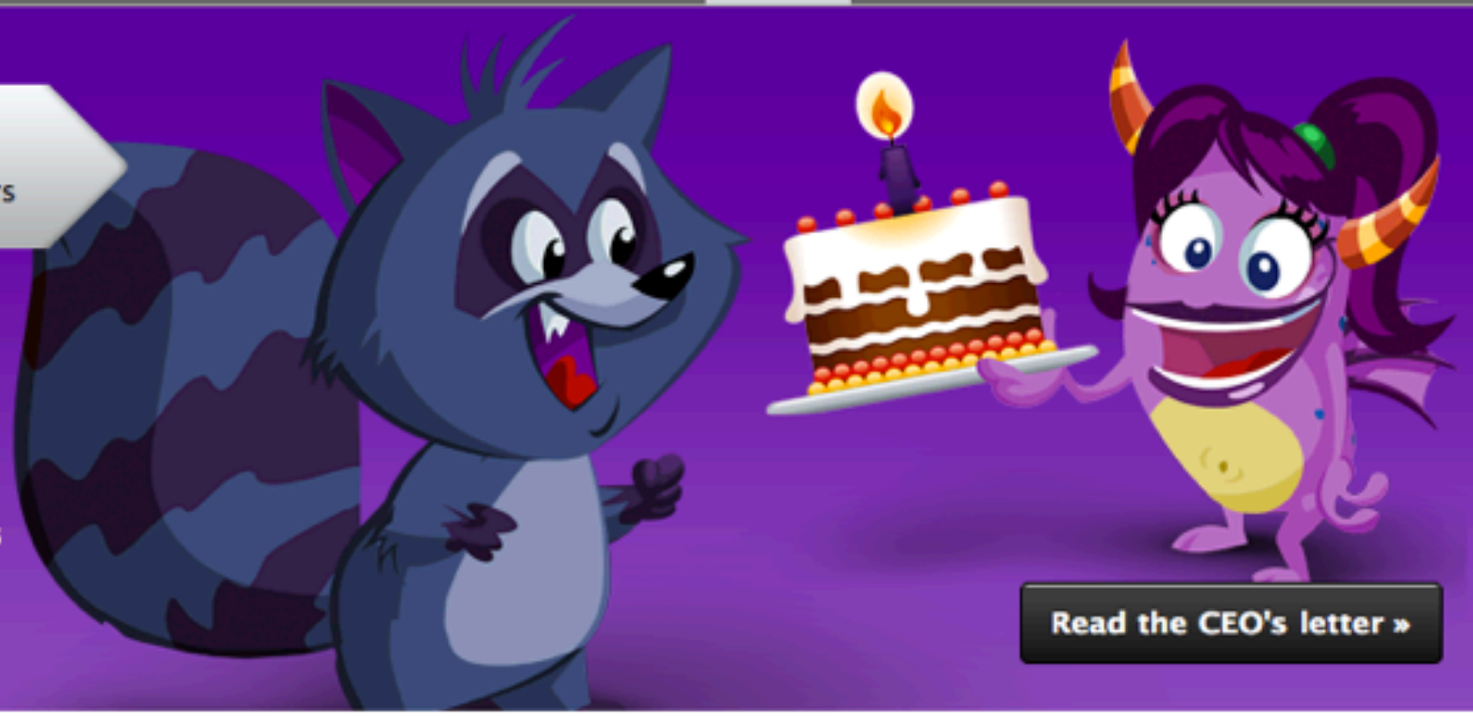
wooga
world of gaming

Home | Games | Jobs | About | Support

wooga's 1st birthday
Celebrating 10 million monthly users

Monster World
Choose your monster family

Bubble Island
Top puzzle game played by millions



Read the CEO's letter »

wooga on Facebook

Like

wooga German start-up blog deutsche-startups.de visited us and published photos of their tour.
How do you like our office in Berlin?

Hausbesuch bei wooga :: deutsche-startups.de
www.deutsche-startups.de
Deutsche-startups - eine Art Infoservice fur die Web2.0-Branche. Hausbesuch bei wooga

Friday at 1:16am

wooga We added interviews from a few of our employees to find out what they like about working at wooga. Check it out:

Based in Berlin, wooga is the leading European social games developer.

We are hiring!

Currently, we are searching for:

- Senior Creative Producer (m/f)
- Vorstandsassistent (m/w) (m/w)
- Software Engineer - Graduate Position (m/f)

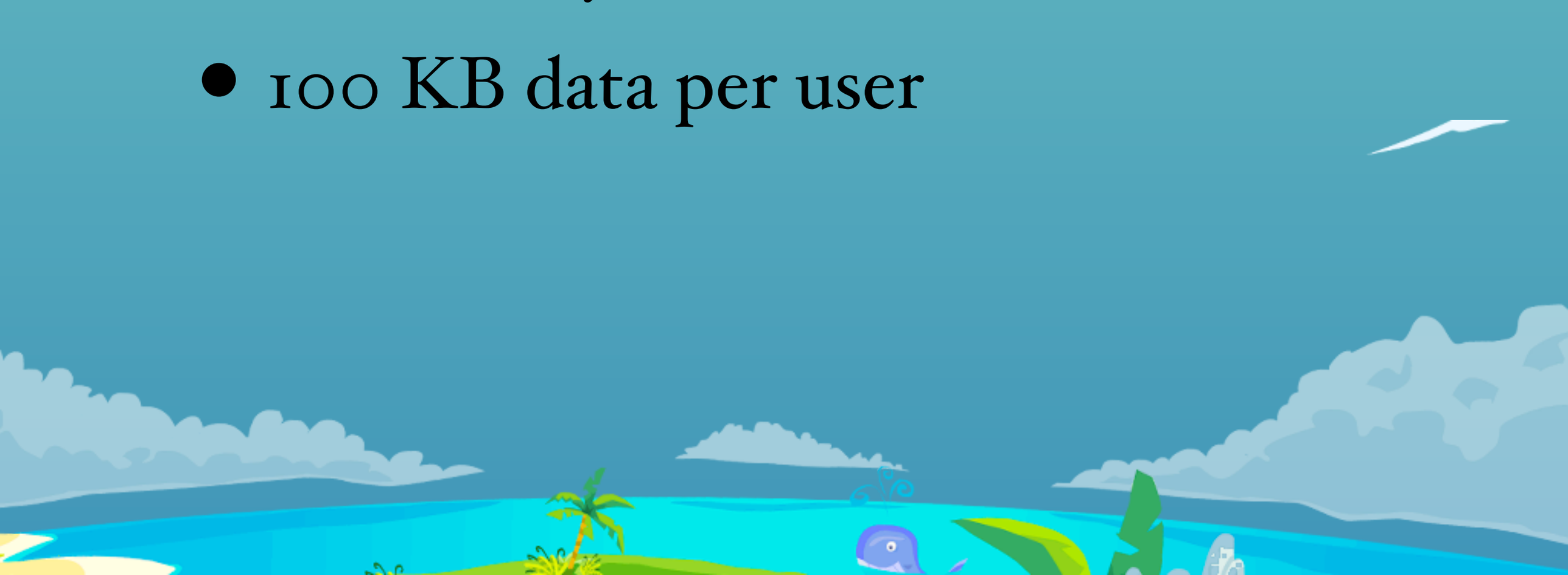
Challenge

Requirements

- backend for facebook game

Requirements

- backend for facebook game
- 1 mio. daily users
- 100 KB data per user



Requirements

- peak traffic:
  - 10.000 concurrent users
  - 3.000 requests/second

Requirements

- peak traffic:
  - 10.000 concurrent users
  - 3.000 requests/second
- write-heavy workload

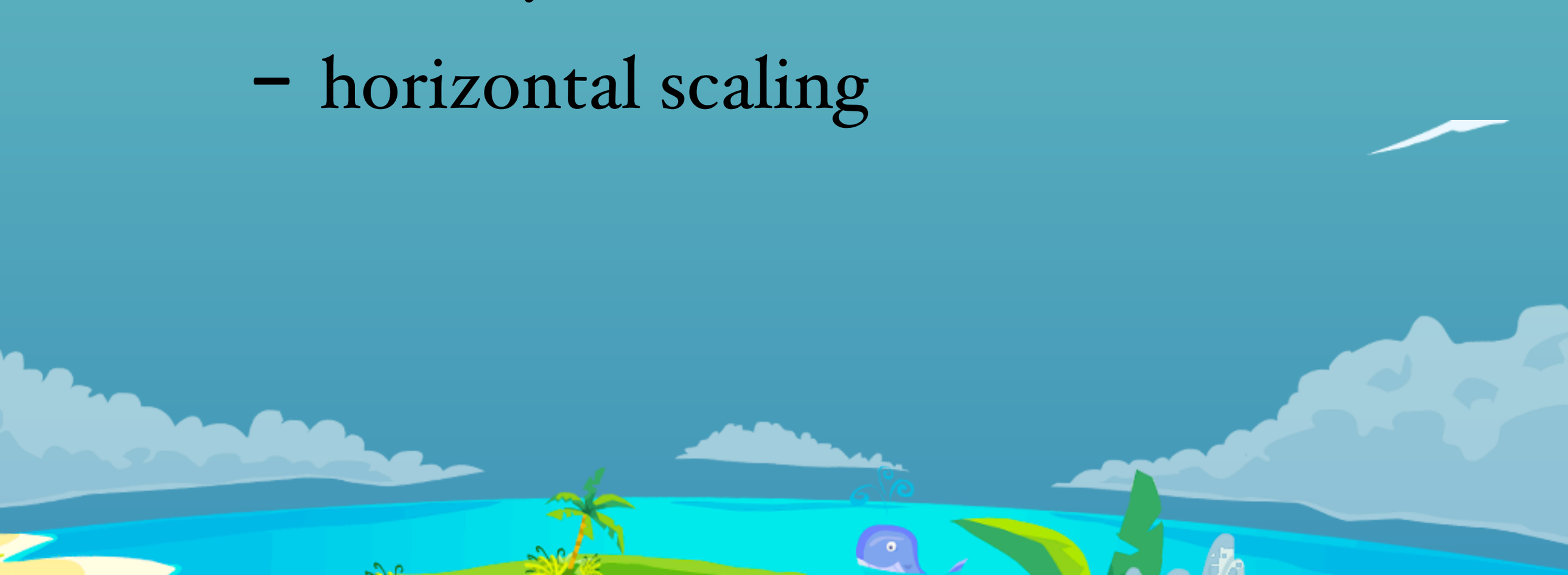
Relational Database

Scaling Reads

- caching (memcached)

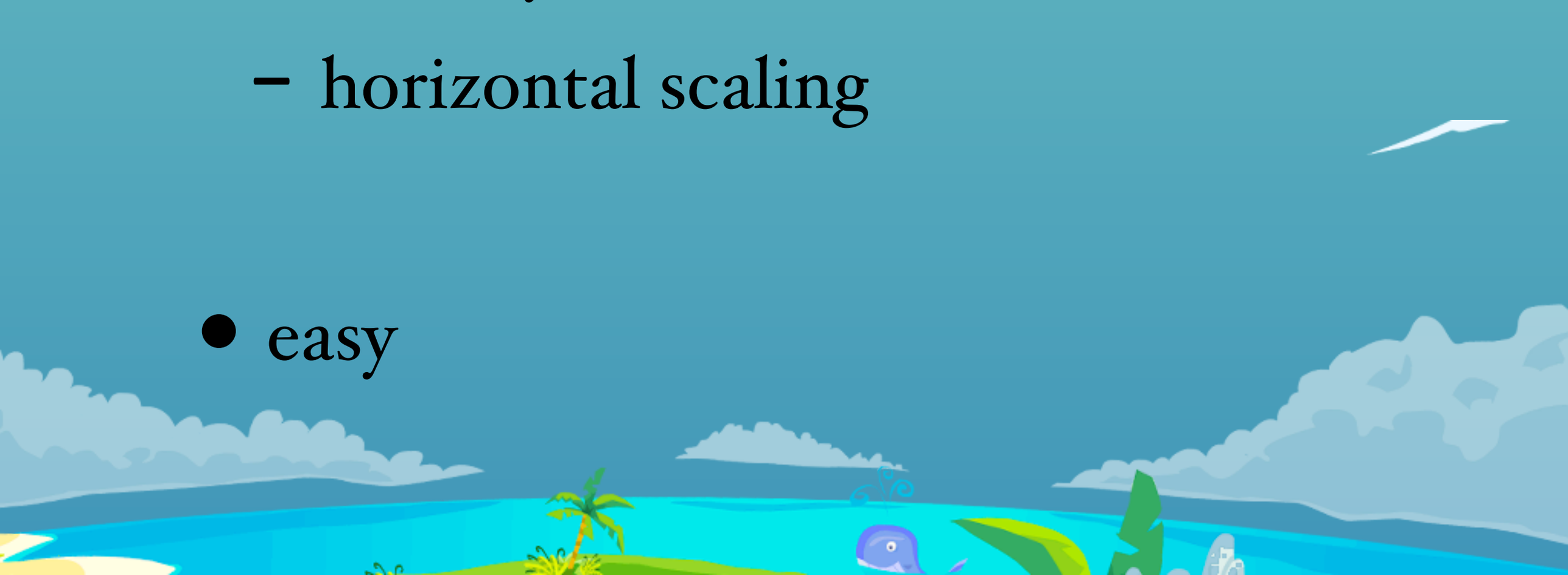
Scaling Reads

- caching (memcached)
- read-only slaves
- horizontal scaling



Scaling Reads

- caching (memcached)
- read-only slaves
  - horizontal scaling
- easy

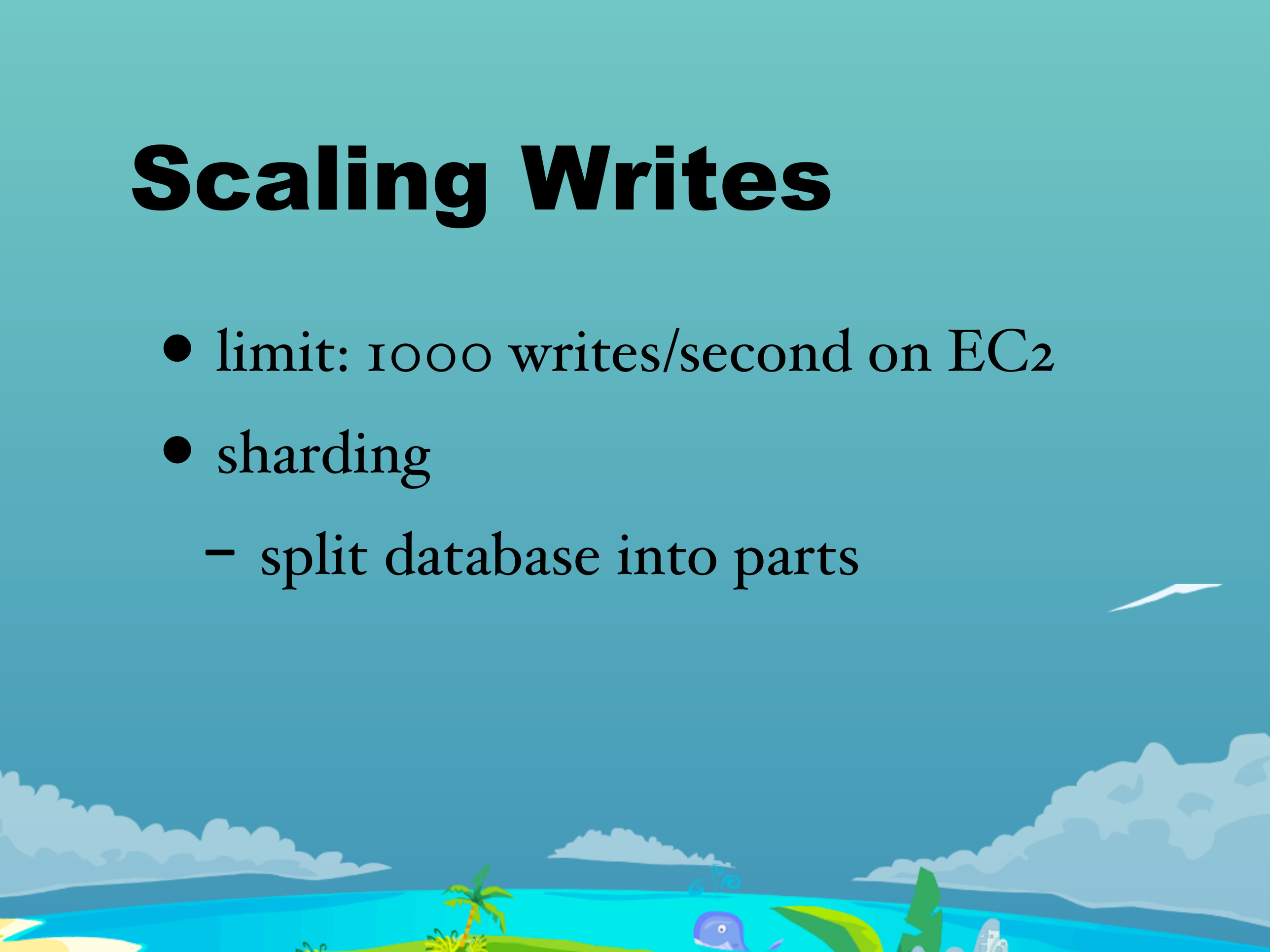


Scaling Writes

- limit: 1000 writes/second on EC2

Scaling Writes

- limit: 1000 writes/second on EC2
- sharding
  - split database into parts



Scaling Writes

- limit: 1000 writes/second on EC2
- sharding
  - split database into parts
- difficult

Alternatives?

Alternatives?

MongoDB

Cassandra

Hypertable

BigTable

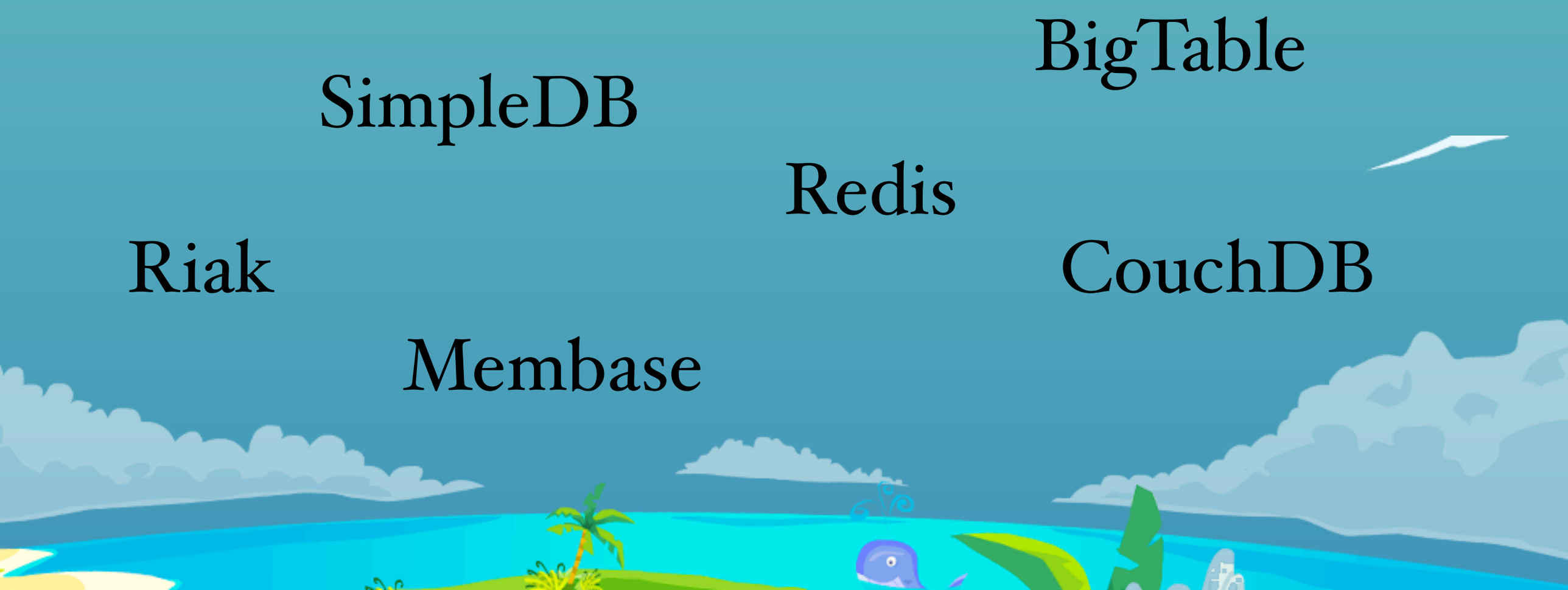
SimpleDB

Redis

Riak

CouchDB

Membase



Basics

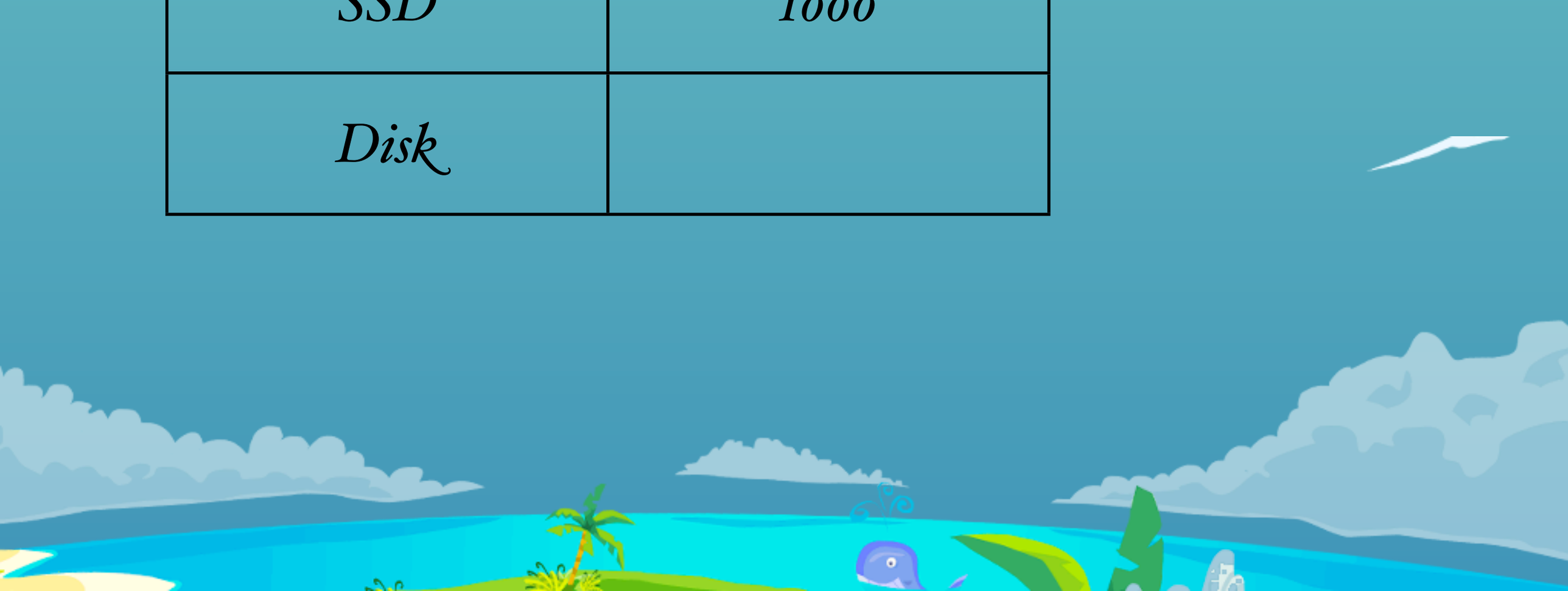
Relative Latency

Memory	*1*
SSD	
Disk	

Relative Latency

Memory	*1*
SSD	*1000*
Disk	

x 1000

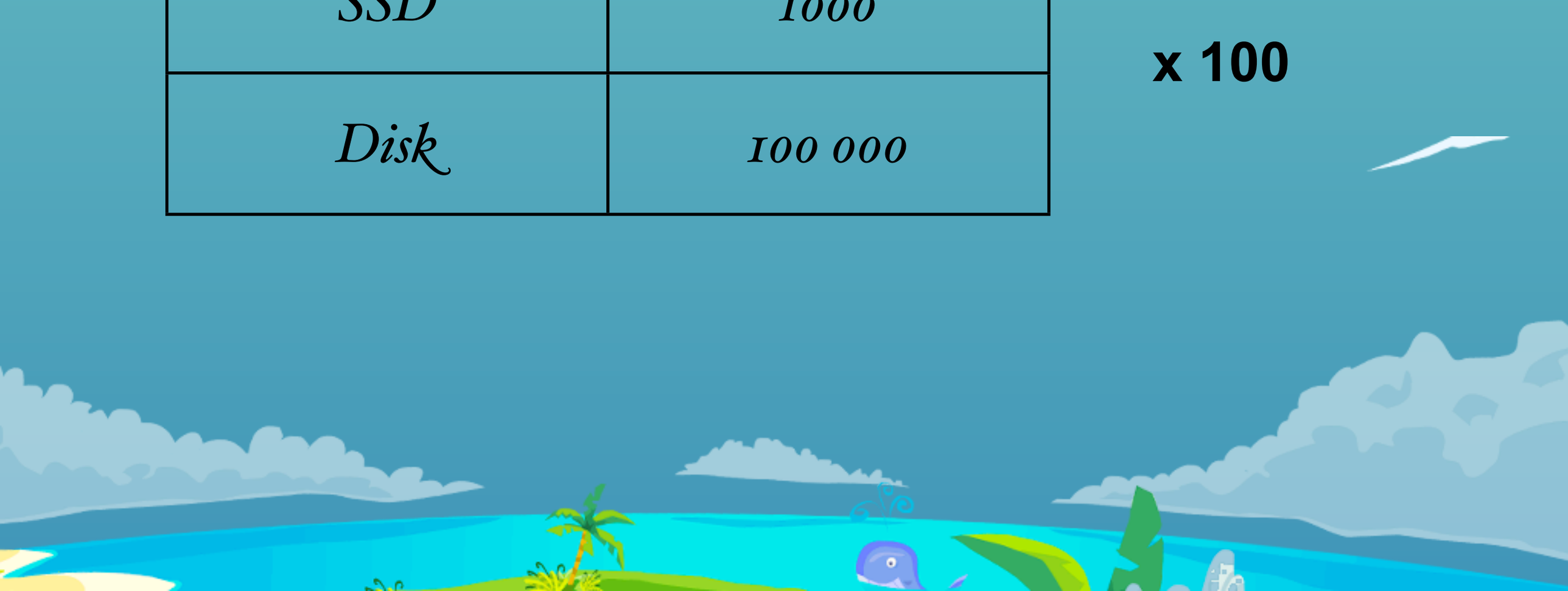


Relative Latency

Memory	*1*
SSD	*1000*
Disk	*100 000*

x 1000

x 100



Pizza Delivery Time

Memory	***30 minutes***
SSD	***3 weeks***
Disk	***5.5 years***

x 1000

x 100

"Memory is the new disk,
disk is the new tape."

—Jim Gray

Rule of thumb

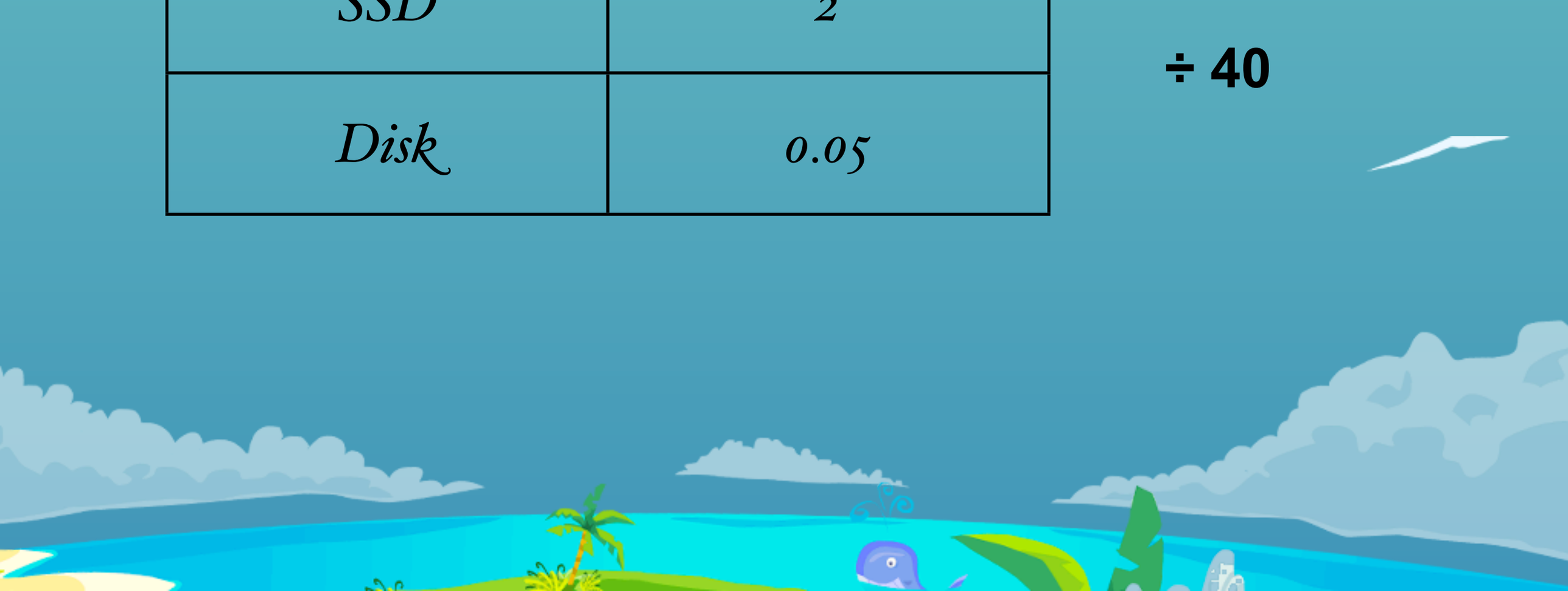
- random access = memory
- sequential access = disk

Euro per GB

Memory	*20*
SSD	*2*
Disk	*0.05*

÷ 10

÷ 40



Redis

Architecture

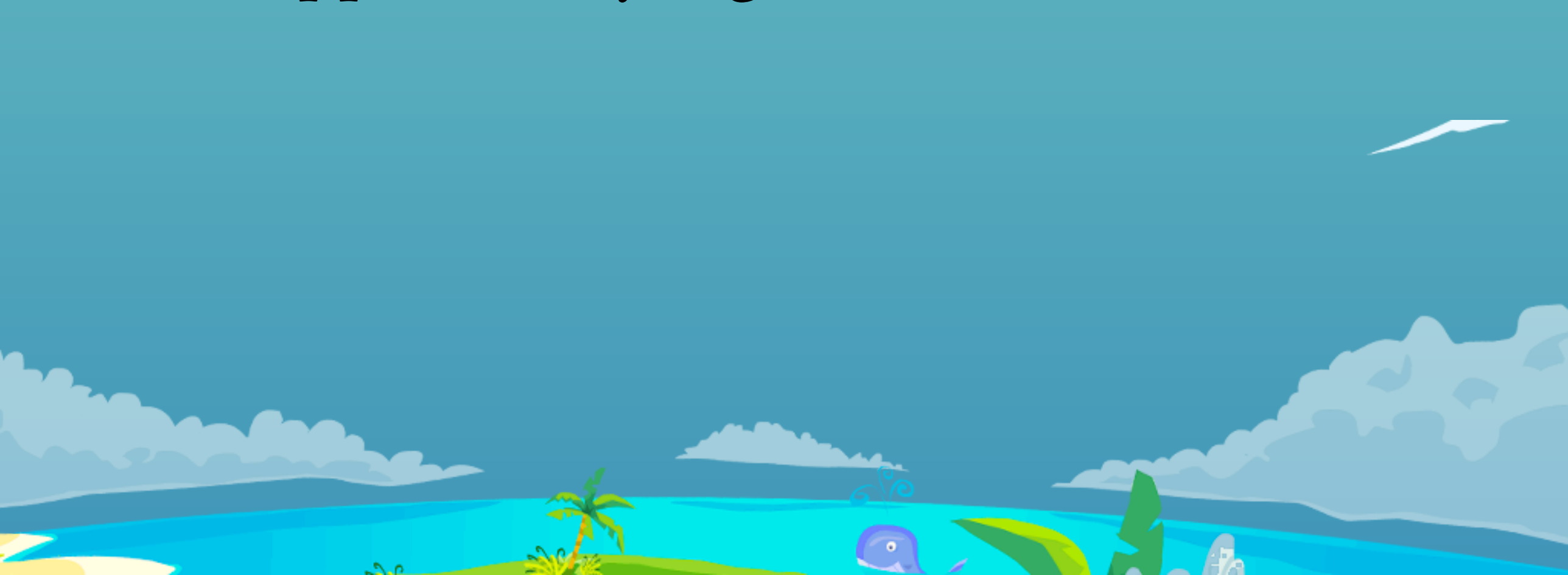
- key-value-store
- in-memory database
  - with virtual memory

Durability

- full database dumps

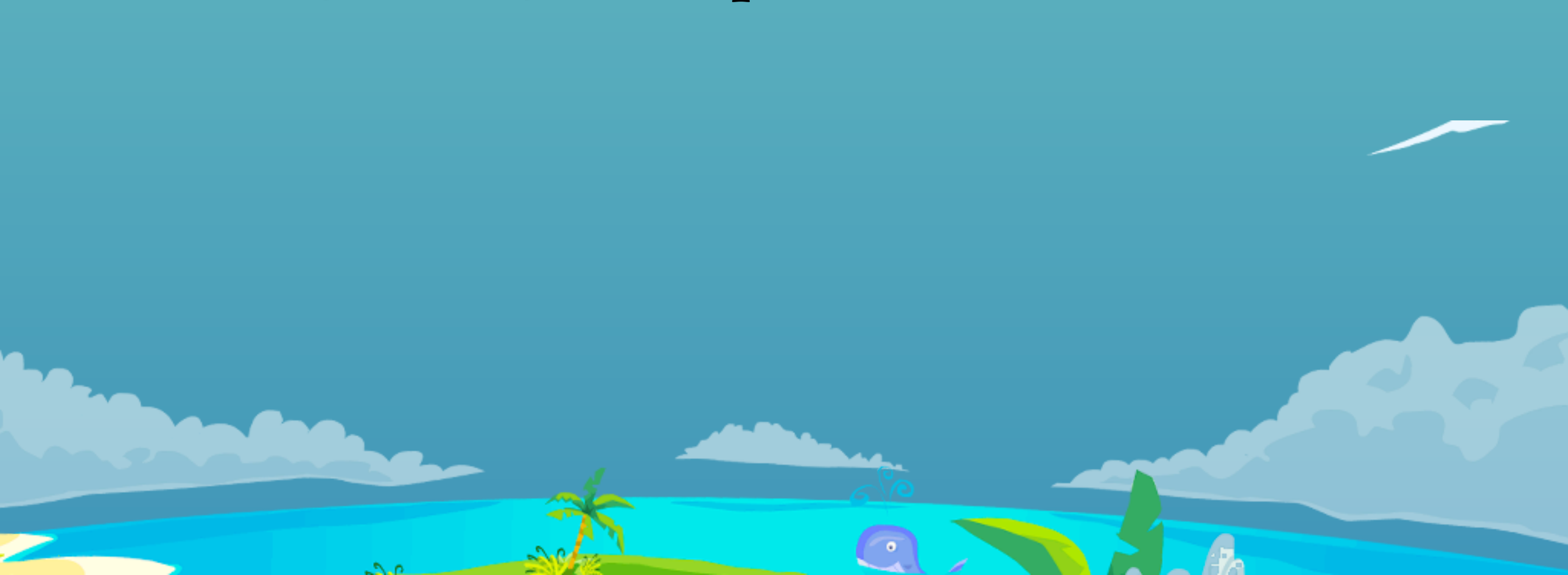
Durability

- full database dumps
- append-only log



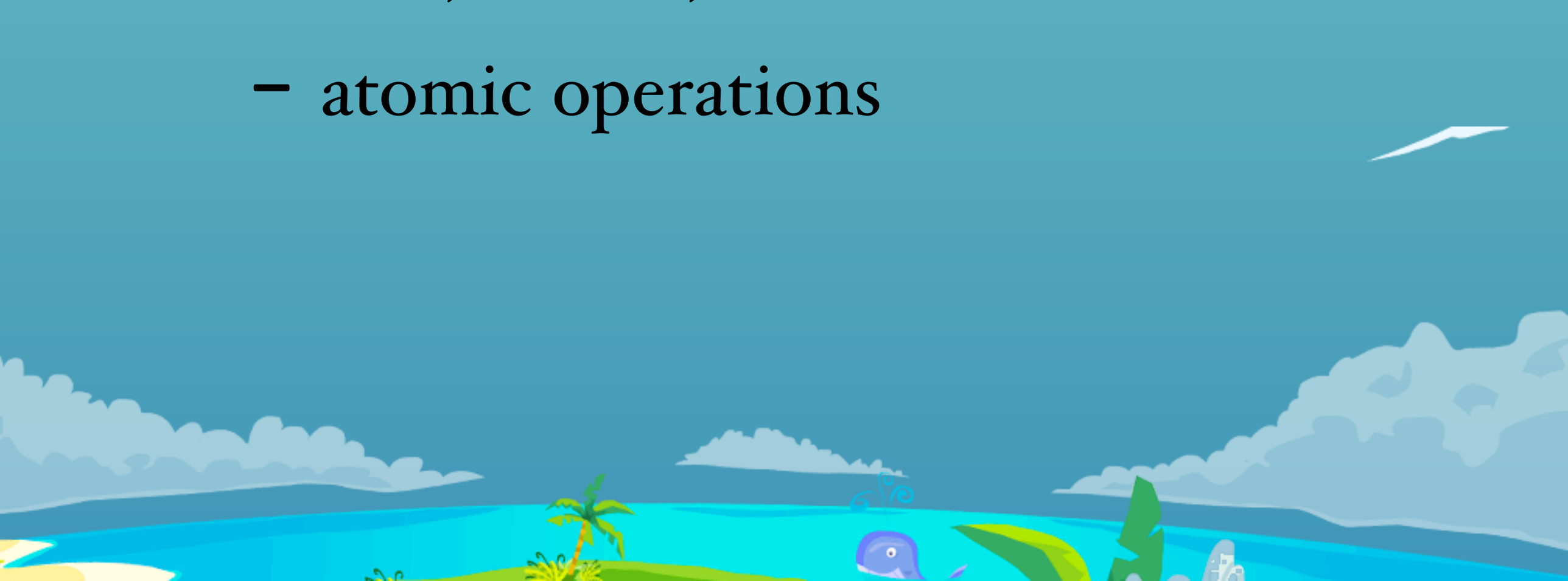
Killer Feature

- high (write) throughput
  - 30 to 150 K operations / second



Other Features

- interesting data structures
  - lists, hashes, (sorted) sets
  - atomic operations



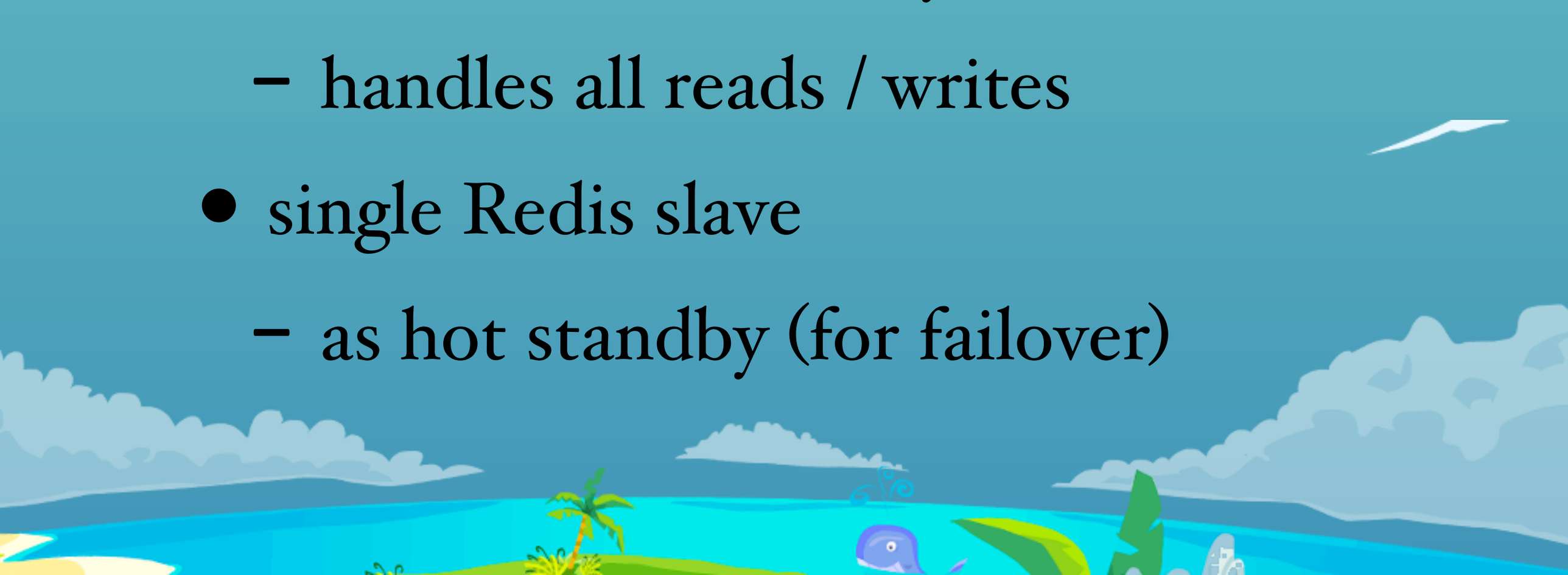
Our Setup

Architecture

- single Redis master
  - with virtual memory
  - handles all reads / writes

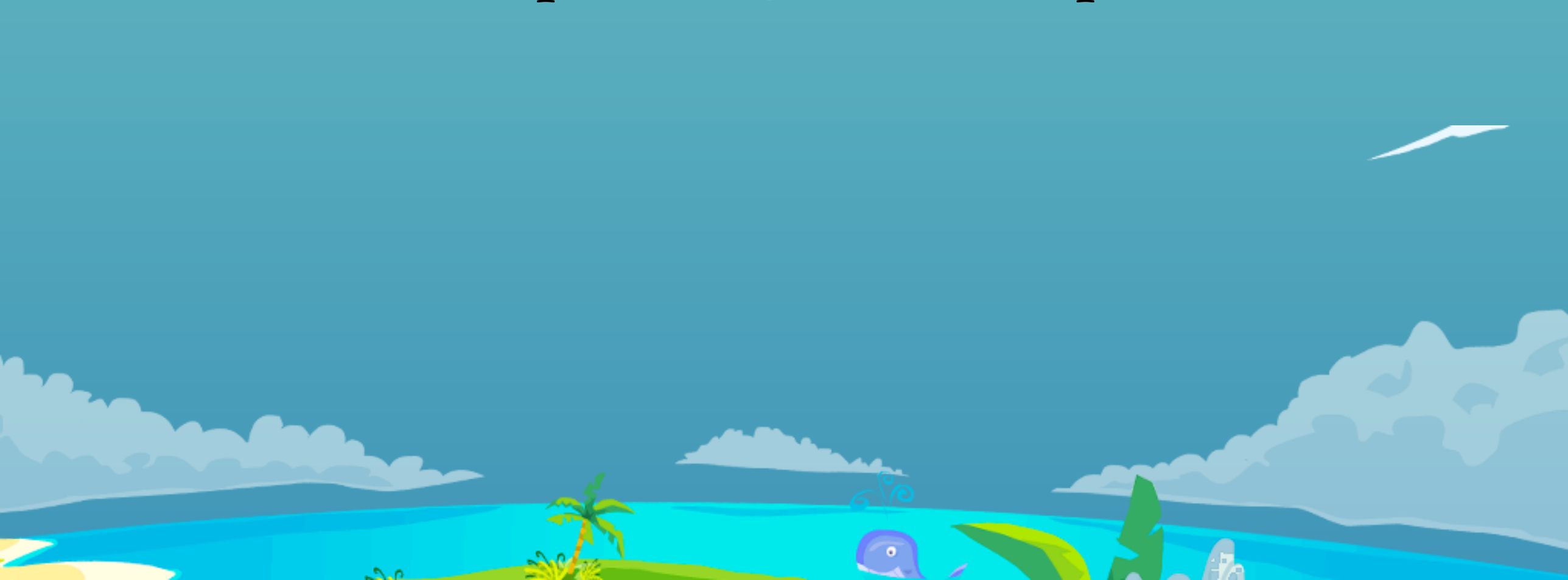
Architecture

- single Redis master
  - with virtual memory
  - handles all reads / writes
- single Redis slave
  - as hot standby (for failover)



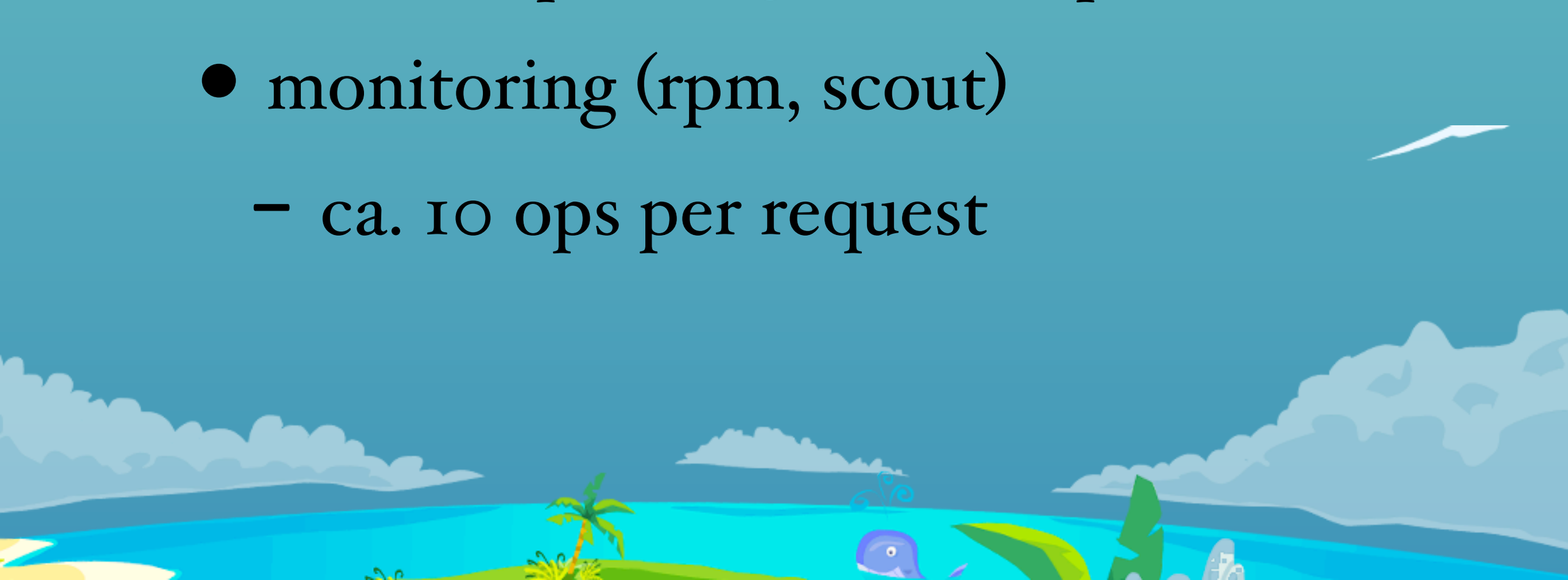
Throughput

- redis-benchmark
  - 60 K ops / s = **3.6 mio** ops / m



Throughput

- redis-benchmark
  - 60 K ops / s = **3.6 mio** ops / m
- monitoring (rpm, scout)
  - ca. 10 ops per request



Throughput

- redis-benchmark
  - 60 K ops / s = **3.6 mio** ops / m
- monitoring (rpm, scout)
  - ca. 10 ops per request

- 200 K req / m = **2.0 mio** ops / m ✔

Capacity 1

- 100 KB / user (on disk)
- 10.000 concurrent users (peak)

Capacity 1

- 100 KB / user (on disk)
- 10.000 concurrent users (peak)
- 1 GB memory
  - (plus Redis overhead) ✔

Capacity 2

- Redis keeps all keys in memory
- 10 mio. total users
- 20 GB / 100 mio. integer keys

Capacity 2

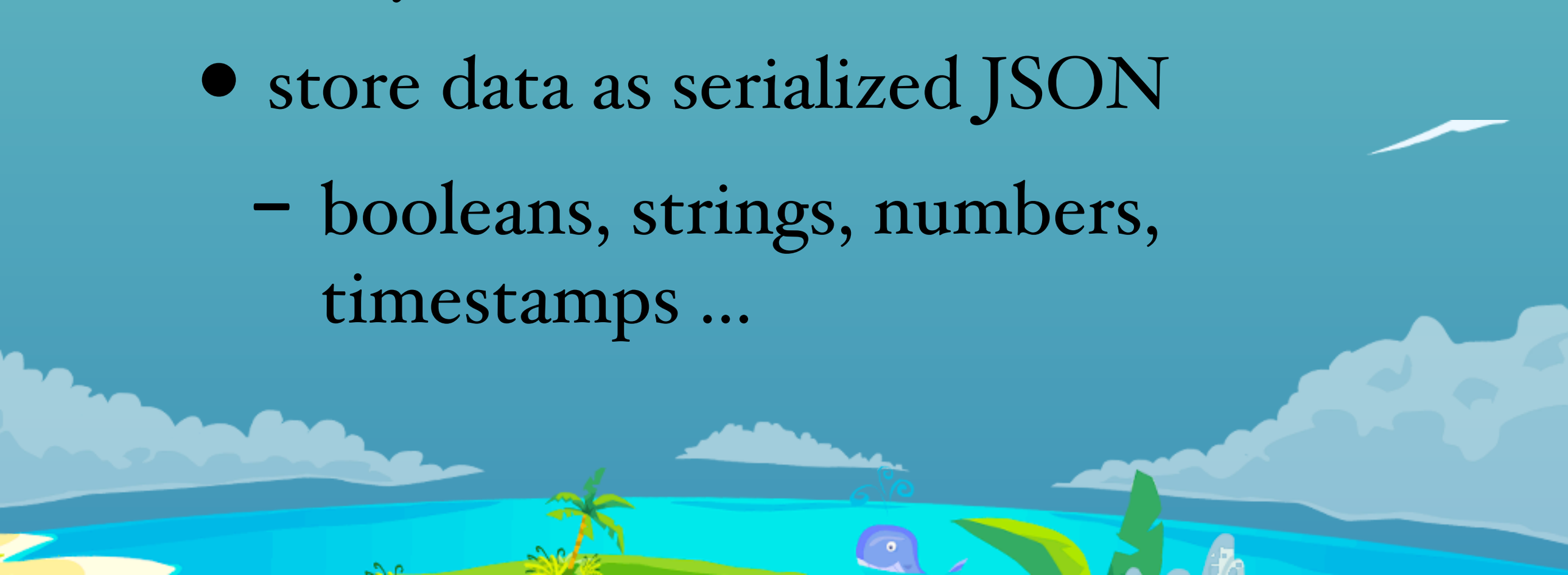
- Redis keeps all keys in memory
- 10 mio. total users
- 20 GB / 100 mio. integer keys
- 2 GB memory for keys ✔

Data model

- one Redis hash per user
  - key: facebook id

Data model

- one Redis hash per user
  - key: facebook id
- store data as serialized JSON
  - booleans, strings, numbers, timestamps ...



Advantages

- efficient to swap user data in / out

Advantages

- efficient to swap user data in / out
- turns Redis into “document db”
  - atomic ops on parts

Advantages

- efficient to swap user data in / out
- turns Redis into “document db”
  - atomic ops on parts
- easy to dump / restore user data

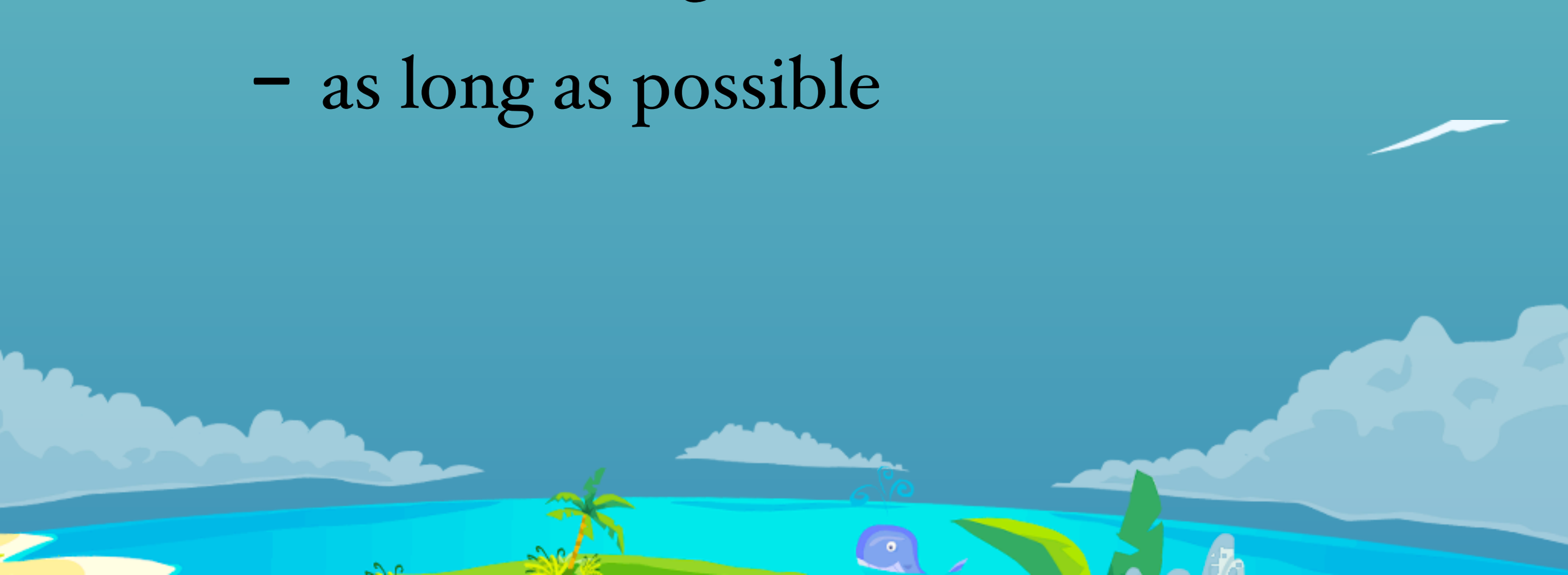
Advice

Advice

- use the right tool for the job

Advice

- use the right tool for the job
- avoid sharding
  - as long as possible



Advice

- use the right tool for the job
- avoid sharding
  - as long as possible
- **keep it simple**

Q & A

Nosql Night Berlin

November 17, 19:30
newthinking store
Tucholskystr. 48

Links

- *"A Conversation with Jim Gray"*
- redis.io
- tim.lossen.de
- wooga.com/jobs