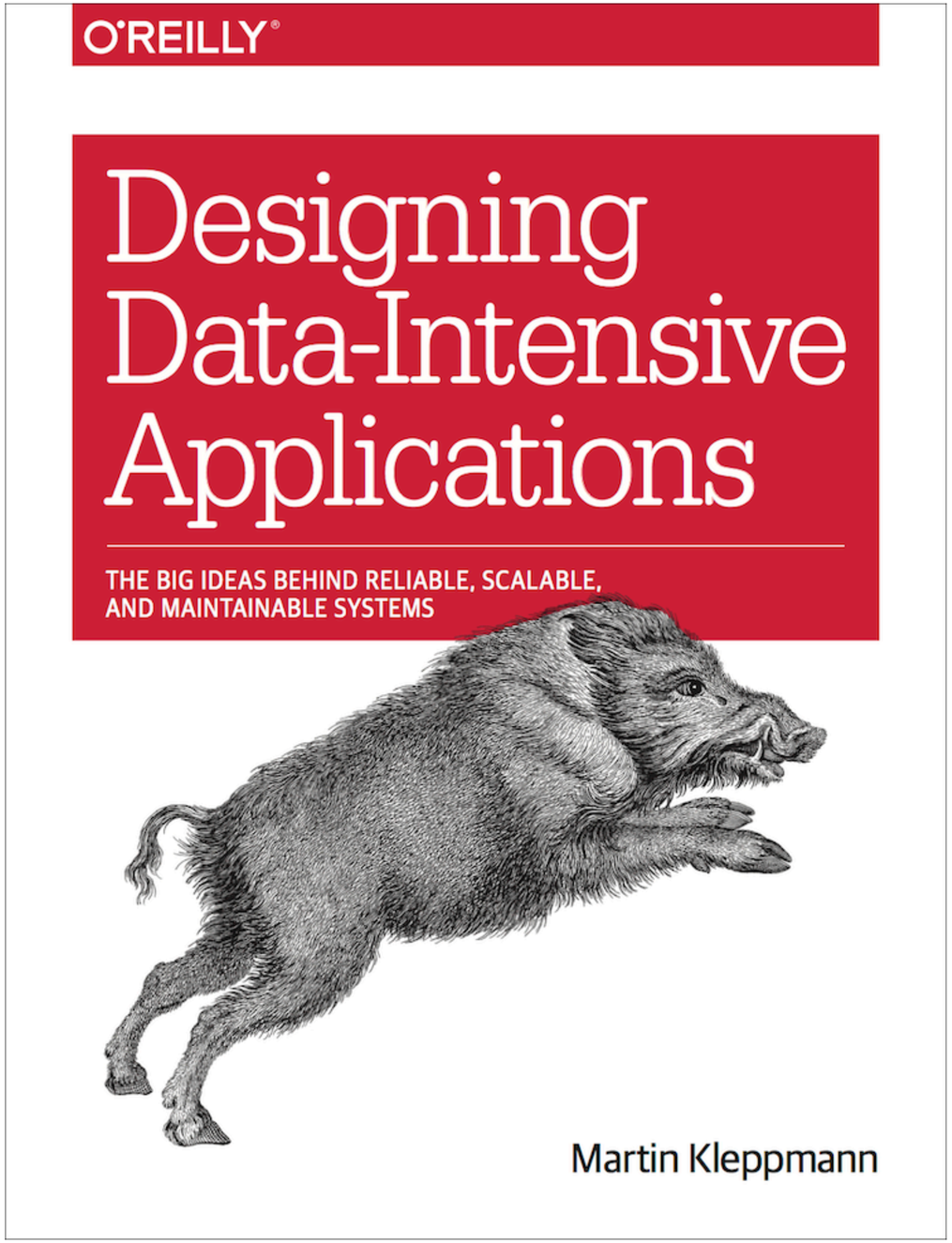
Reading Club

Zhiyi Xiao



Part I. Foundations of Data Systems

- 1. Reliable, Scalable, and Maintainable Applications
- 2. Data Models and Query Languages.
- 3. Storage and Retrieval.

About Data-intensive application

- Data-intensive application
  - data is its primary challenge
  - the quantity of data, the complexity of data, or the speed at which it is changing
- Compute-intensive
  - CPU cycles are the bottleneck.

After reading this book

- will be in a great position to decide which kind of technology is appropriate for which purpose
- understand how tools can be combined to form the foundation of a good application architecture
- develop a good intuition for what your systems are doing under the hood so that you can reason about their behavior,
- make good design decisions, and track down any problems that may arise.

Who Should Read This Book

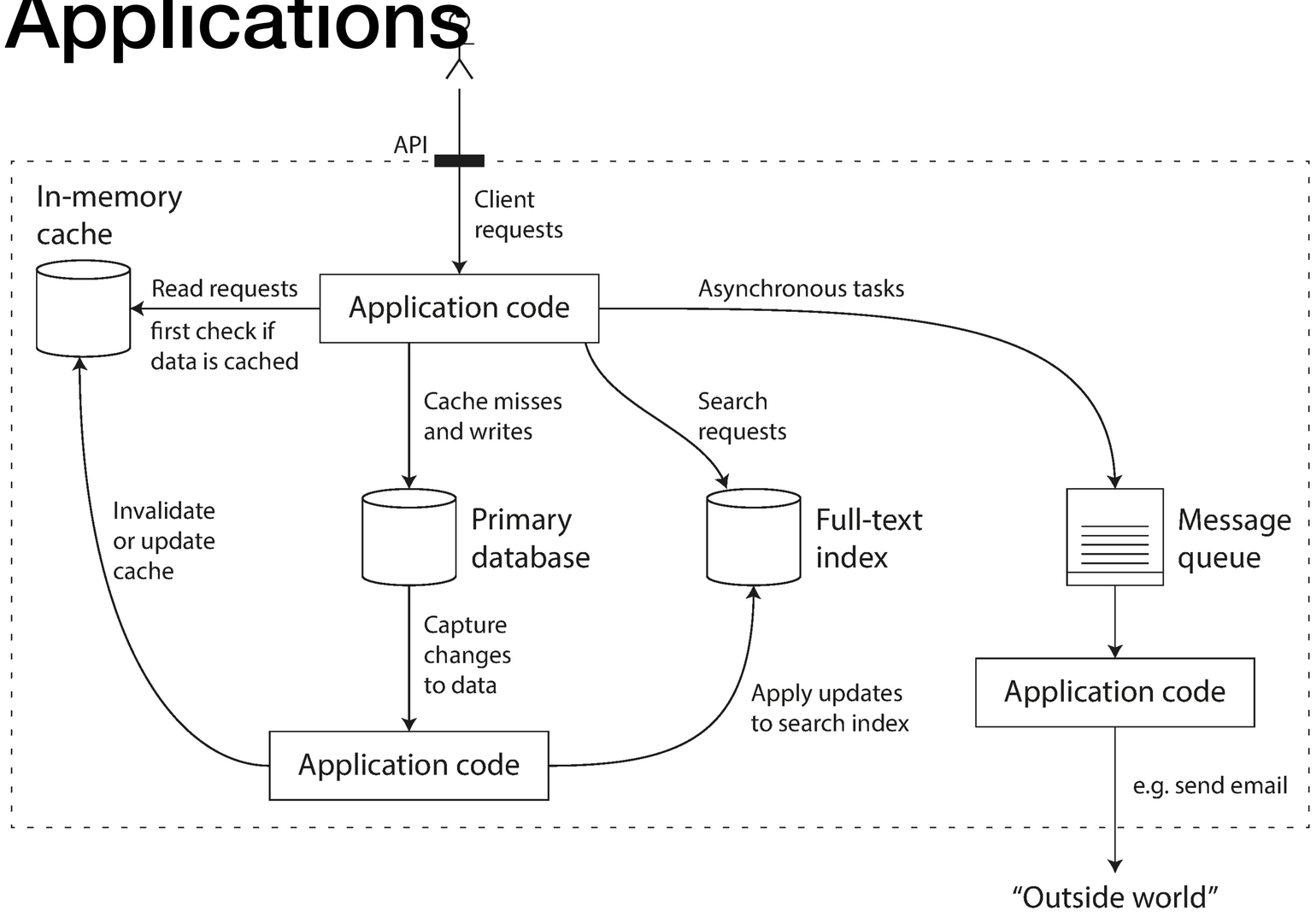
- want to learn how to make data systems scalable
- need to make applications highly available
- are looking for ways of making systems easier to maintain in the long run
- have a natural curiosity for the way things work

Outline of This Book

- In Part I, we discuss the fundamental ideas that underpin the design of dataintensive applications.
- In Part II, we move from data stored on one machine to data that is distributed across multiple machines.
- In Part III, we discuss systems that derive some datasets from other datasets.

Ch1. Reliable, Scalable, and Maintainable Applications

- database
- caches
- search indexes
- stream processing
- batch processing



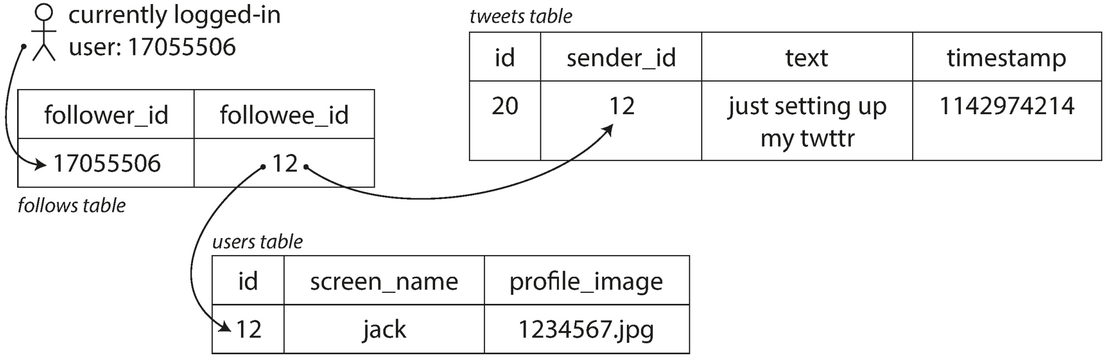
Reliability

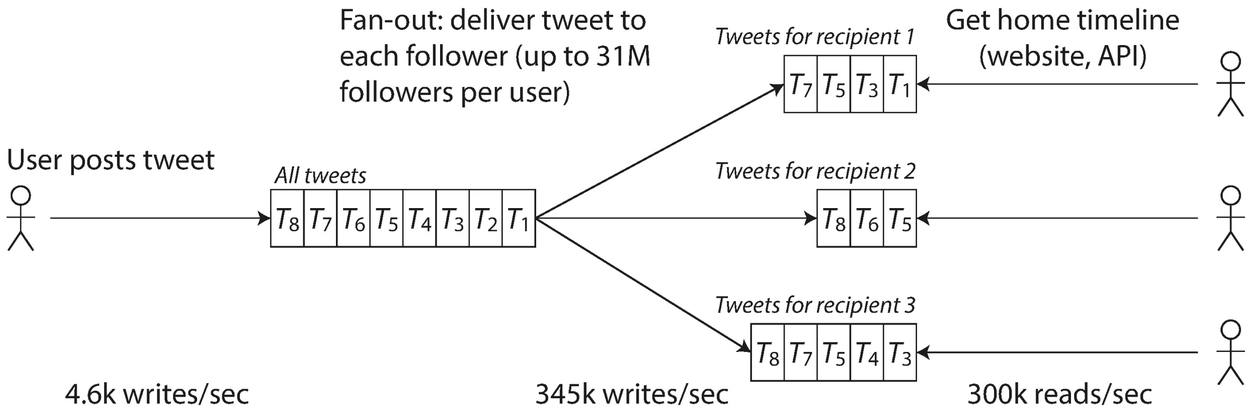
- “continuing to work correctly, even when things go wrong.”
- Faults
  - fault-tolerant or resilient
  - Hardware Faults: MTTF of about 10 to 50 years
  - Software Errors
  - Human Errors: Tencent Cloud

Scalability

- to describe a system’s ability to cope with increased load.

Twitter home timeline

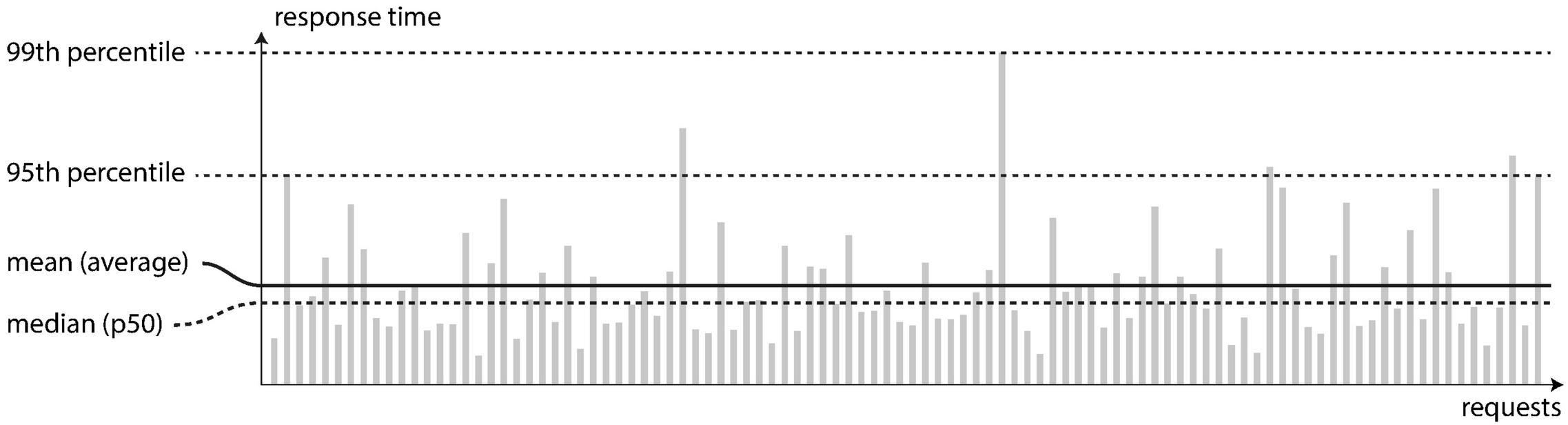




Describing Performance

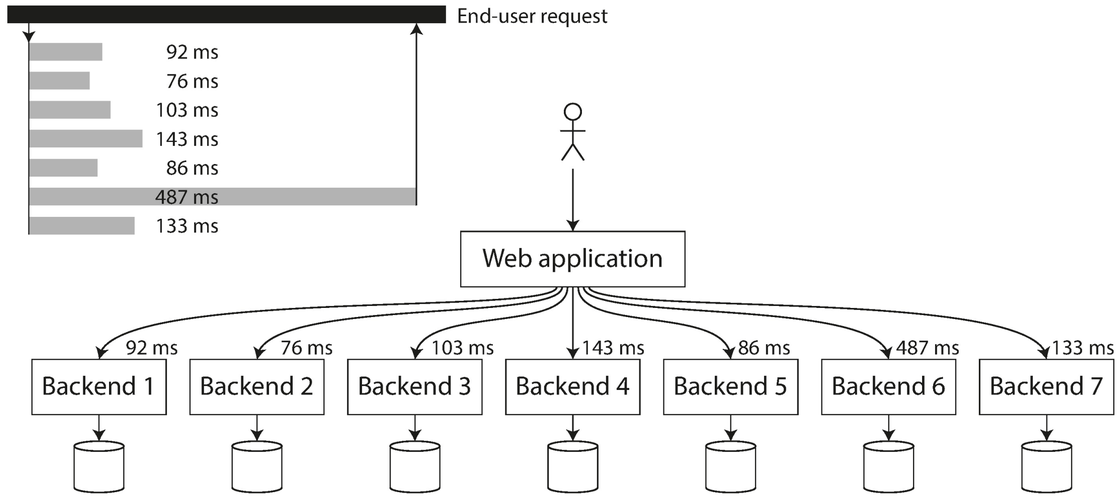
- throughput
  - the number of records we can process per second
  - the total time it takes to run a job on a dataset of a certain size.
- response time
  - the time between a client sending a request and receiving a response.

response times



metric

- the median
  - how long users typically have to wait
- tail latencies
  - the 95th, 99th, and 99.9th percentiles
- service level objectives (SLOs)
- <u>service level agreements (SLAs)</u>



Maintainability

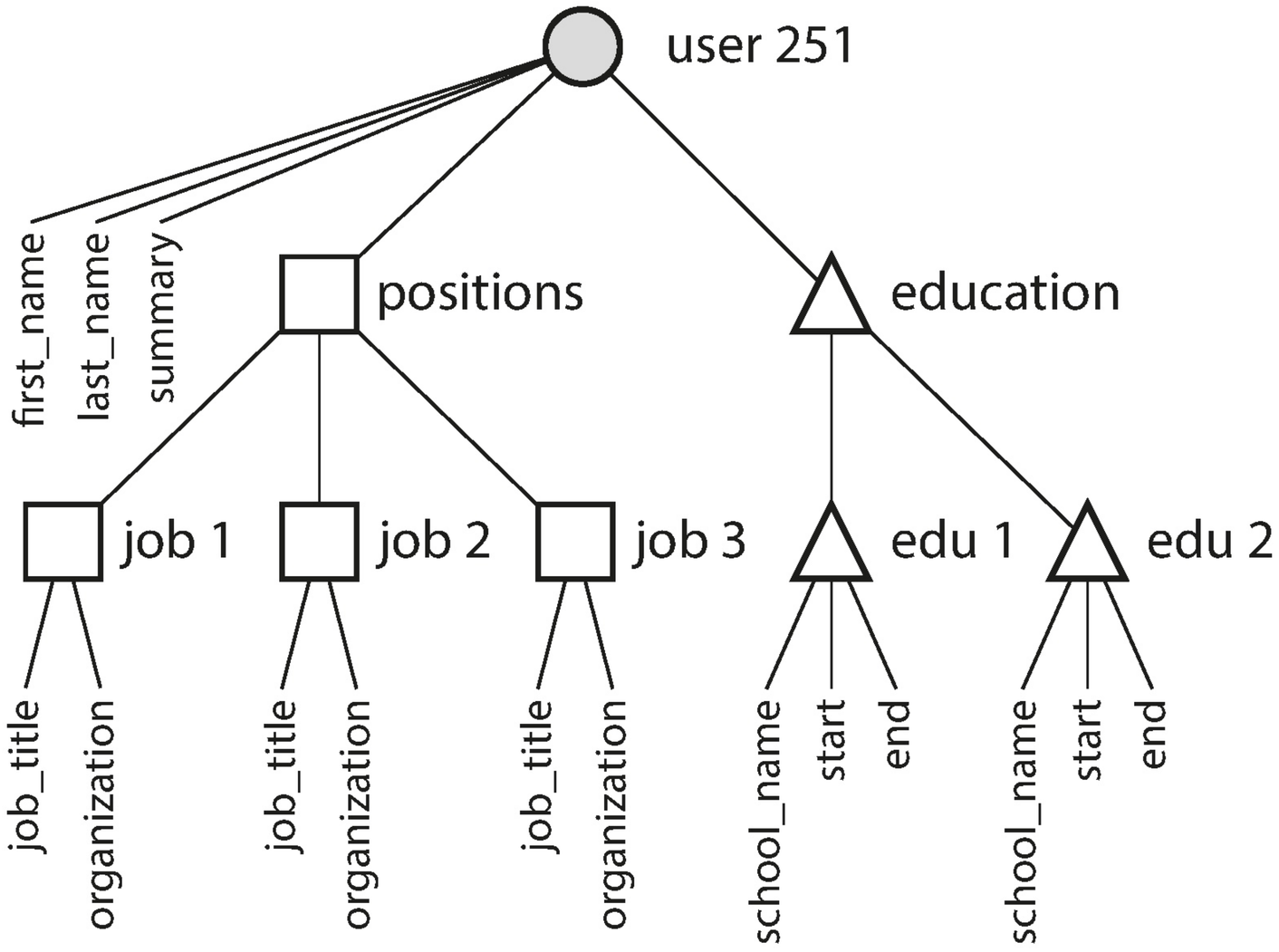
- It is well known that the majority of the cost of software is not in its initial development, but in its **ongoing maintenance**
- Operability
- Simplicity
- Evolvability

Ch2. Data Models and Query Languages

- the relational model
- the document model
- graph-based data models
- query languages

```
{
  "user_id":     251,
  "first_name":  "Bill",
  "last_name":   "Gates",
  "summary":     "Co-chair of the Bill & Melinda Gates... Active blogger.",
  "region_id":   "us:91",
  "industry_id": 131,
  "photo_url":   "/p/7/000/253/05b/308dd6e.jpg",

  "positions": [
    {"job_title": "Co-chair", "organization": "Bill & Melinda Gates Foundation"},
    {"job_title": "Co-founder, Chairman", "organization": "Microsoft"}
  ],
  "education": [
    {"school_name": "Harvard University",       "start": 1973, "end": 1975},
    {"school_name": "Lakeside School, Seattle", "start": null, "end": null}
  ],
  "contact_info": {
    "blog":    "http://thegatesnotes.com",
    "twitter": "http://twitter.com/BillGates"
  }
}
```

One to many relationships



http://www.linkedin.com/in/williamhgates

Bill Gates

Greater Seattle Area | Philanthropy

Summary

Co-chair of the Bill & Melinda Gates Foundation. Chairman, Microsoft Corporation. Voracious reader. Avid traveler. Active blogger.

Experience

Co-chair • Bill & Melinda Gates Foundation
2000 – Present

Co-founder, Chairman • Microsoft
1975 – Present

Education

Harvard University
1973 – 1975

Lakeside School, Seattle

Contact Info

Blog: thegatesnotes.com
Twitter: @BillGates

users table

user_id	first_name	last_name	summary	region_id	industry_id	photo_id
251	Bill	Gates	Co-chair of … blogger.	us:91	131	57817532

regions table

id	region_name
us:7	Greater Boston Area
us:91	Greater Seattle Area

industries table

id	industry_name
43	Financial Services
48	Construction
131	Philanthropy

positions table

id	user_id	job_title	organization
458	251	Co-chair	Bill & Melinda Gates F…
457	251	Co-founder, Chairman	Microsoft

education table

id	user_id	school_name	start	end
807	251	Harvard University	1973	1975
806	251	Lakeside School, Seattle	NULL	NULL

contact_info table

id	user_id	type	url
155	251	blog	http://thegatesnotes.com
156	251	twitter	http://twitter.com/BillGates

Schema flexibility in the document model

- schema-on-read
- schema-on-write

```
if (user && user.name && !user.first_name) {
    // Documents written before Dec 8, 2013 don't have first_name
    user.first_name = user.name.split(" ")[0];
}
```

```
ALTER TABLE users ADD COLUMN first_name text;
UPDATE users SET first_name = split_part(name, ' ', 1);      -- PostgreSQL
UPDATE users SET first_name = substring_index(name, ' ', 1);      -- MySQL
```

Query Languages for Data

- SQL is a declarative query language
- IMS and CODASYL queried the database using imperative code

```
function getSharks() {
    var sharks = [];
    for (var i = 0; i < animals.length; i++) {
        if (animals[i].family === "Sharks") {
            sharks.push(animals[i]);
        }
    }
    return sharks;
}
```

$$\text{sharks} = \sigma_{\text{family} = \text{“Sharks”}} (\text{animals})$$

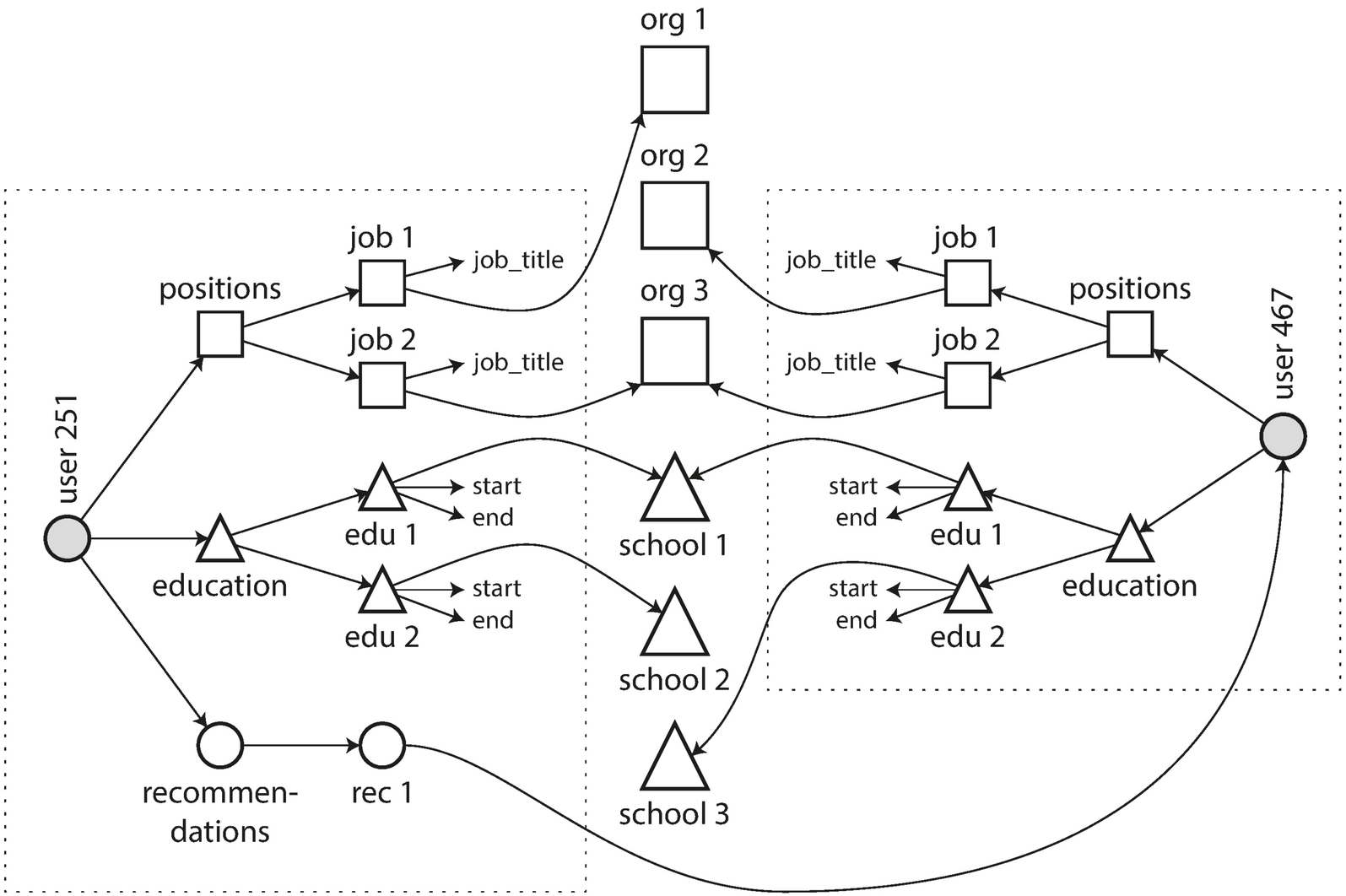
```
SELECT * FROM animals WHERE family = 'Sharks';
```

MapReduce Querying

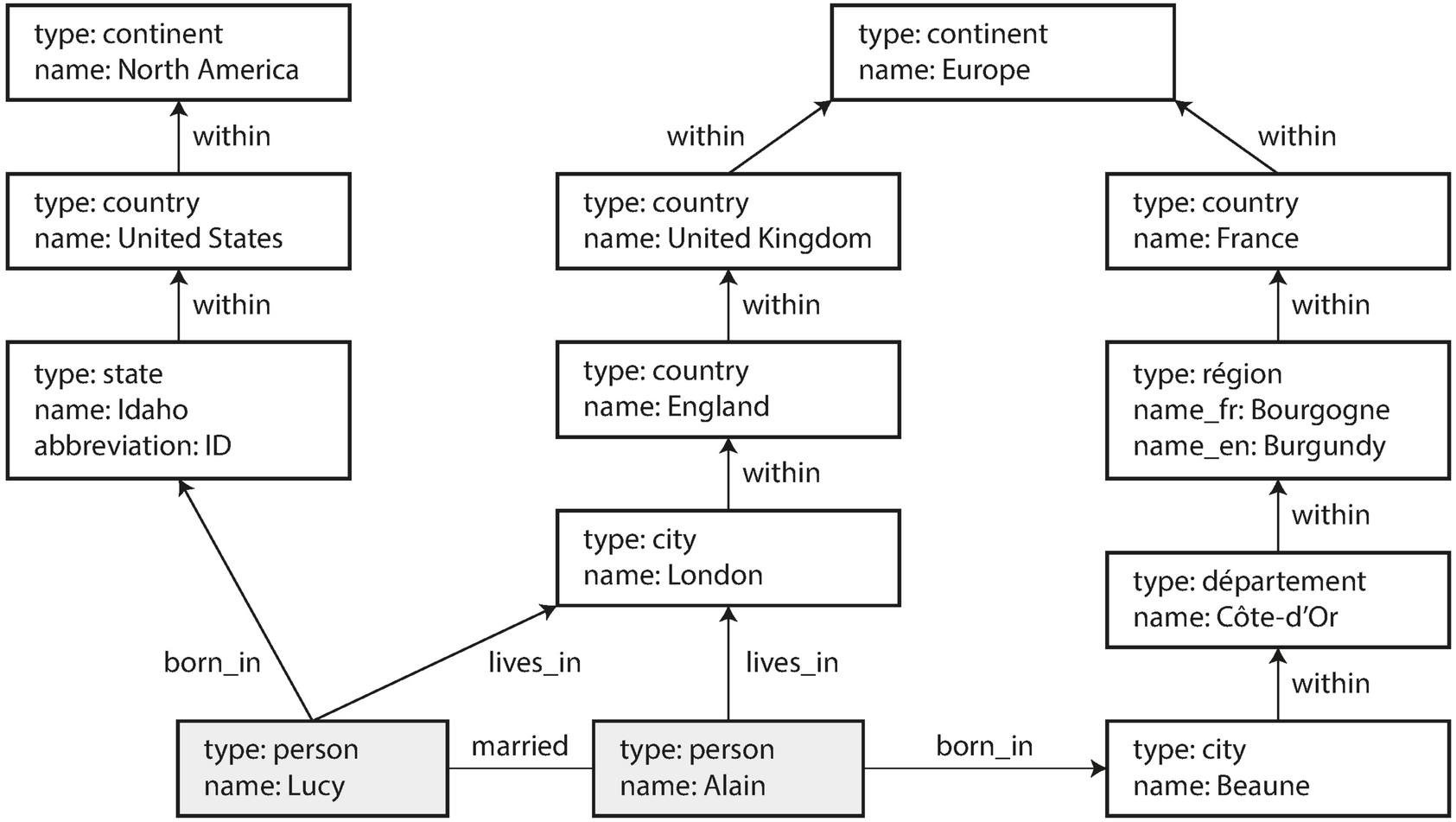
```sql
SELECT date_trunc('month', observation_timestamp) AS observation_month, ❶
       sum(num_animals) AS total_animals
FROM observations
WHERE family = 'Sharks'
GROUP BY observation_month;
```

```
db.observations.mapReduce(
    function map() { ❷
        var year  = this.observationTimestamp.getFullYear();
        var month = this.observationTimestamp.getMonth() + 1;
        emit(year + "-" + month, this.numAnimals); ❸
    },
    function reduce(key, values) { ❹
        return Array.sum(values); ❺
    },
    {
        query: { family: "Sharks" }, ❶
        out: "monthlySharkReport" ❻
    }
);
```

many-to-many relationships



Graph-Like Data Models



The Cypher Query Language

```
CREATE
  (NAmerica:Location {name:'North America', type:'continent'}),
  (USA:Location      {name:'United States', type:'country'  }),
  (Idaho:Location    {name:'Idaho',         type:'state'    }),
  (Lucy:Person       {name:'Lucy' }),
  (Idaho) -[:WITHIN]->  (USA)  -[:WITHIN]-> (NAmerica),
  (Lucy)  -[:BORN_IN]-> (Idaho)
```

```
MATCH
  (person) -[:BORN_IN]->  () -[:WITHIN*0..]-> (us:Location {name:'United States'}),
  (person) -[:LIVES_IN]-> () -[:WITHIN*0..]-> (eu:Location {name:'Europe'})
RETURN person.name
```

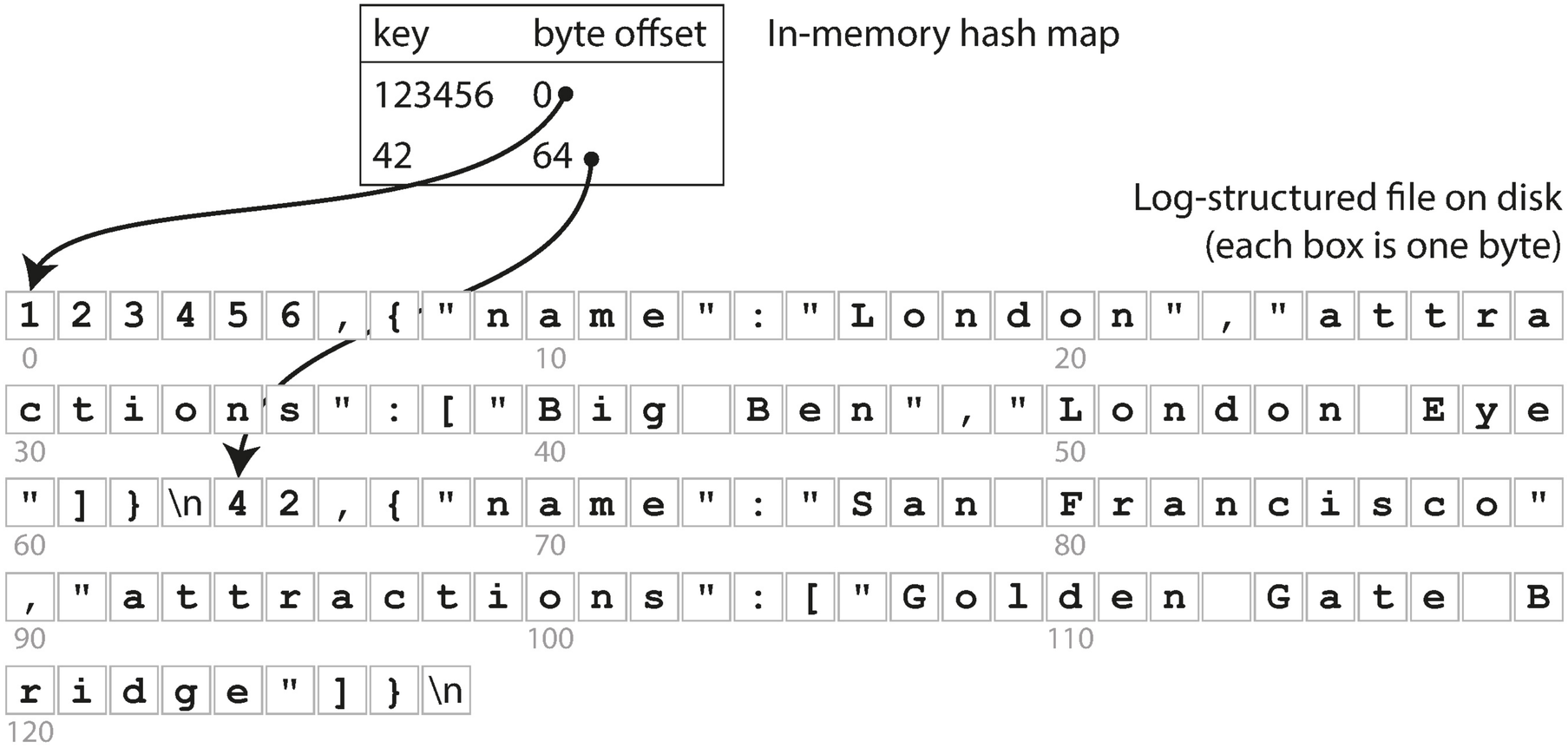
Ch3. Storage and Retrieval

- storage engines
  - log-structured storage engines,
  - page-oriented storage engines

```bash
#!/bin/bash

db_set () {
    echo "$1,$2" >> database
}

db_get () {
    grep "^$1," database | sed -e "s/^$1,//" | tail -n 1
}
```

```
$ db_set 123456 '{"name":"London","attractions":["Big Ben","London Eye"]}'

$ db_set 42 '{"name":"San Francisco","attractions":["Golden Gate Bridge"]}'

$ db_get 42
{"name":"San Francisco","attractions":["Golden Gate Bridge"]}
```

Hash Indexs


key
byte offset
123456
0
42
64
In-memory hash map
Log-structured file on disk
(each box is one byte)
1 2 3 4 5 6 , { " n a m e " : " L o n d o n " , " a t t r a
0
10
20
c t i o n s " : [" B i g B e n " , " L o n d o n E y e
30
40
50
"] } \n 4 2 , { " n a m e " : " S a n F r a n c i s c o "
60
70
80
, " a t t r a c t i o n s " : [" G o l d e n G a t e B
90
100
110
r i d g e "] } \n
120

Compaction

Data file segment

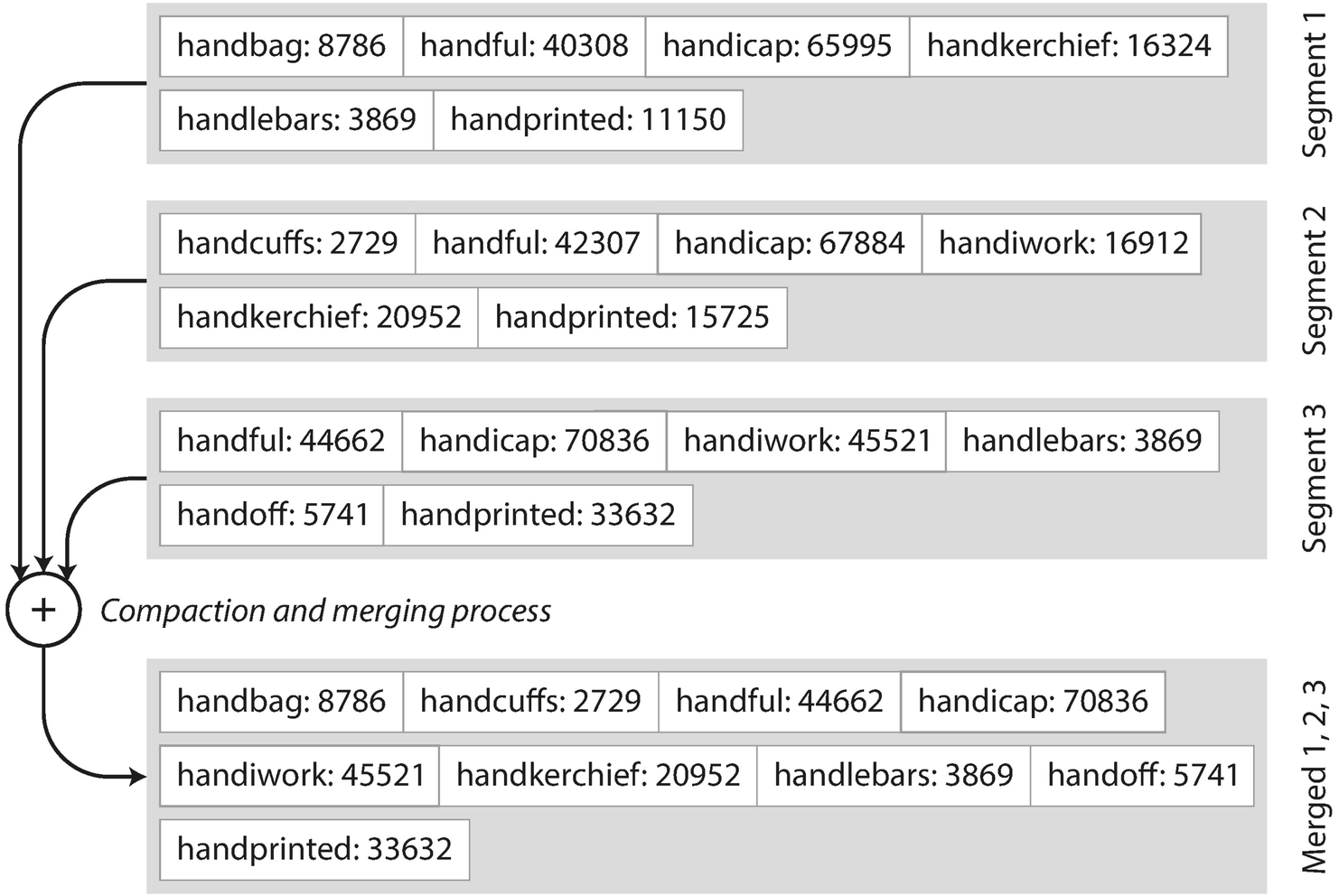
mew: 1078	purr: 2103	purr: 2104	mew: 1079	mew: 1080	mew: 1081
purr: 2105	purr: 2106	purr: 2107	yawn: 511	purr: 2108	mew: 1082

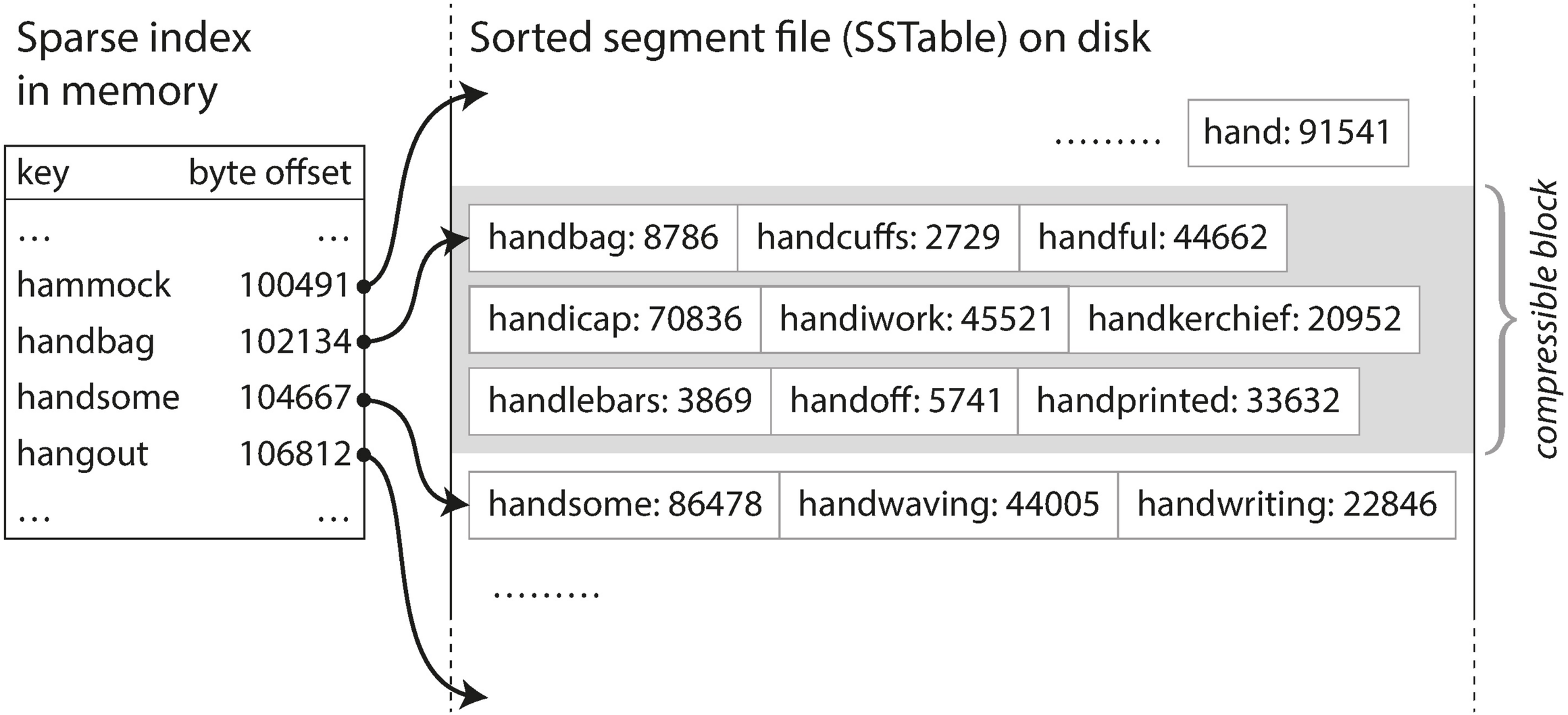
Compaction process

Compacted segment

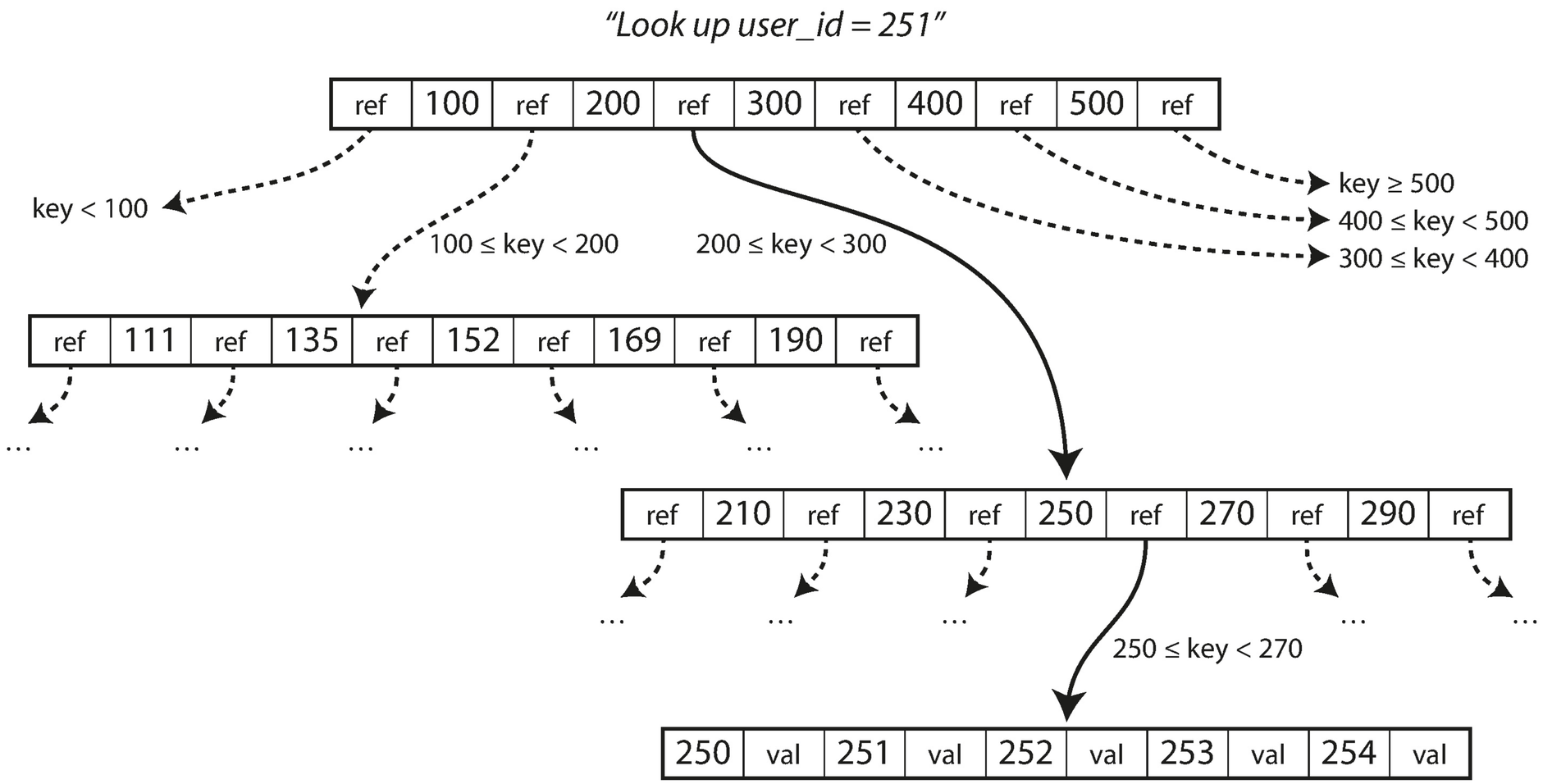
yawn: 511	mew: 1082	purr: 2108

SSTable


handbag: 8786
handful: 40308
handicap: 65995
handkerchief: 16324
handlebars: 3869
handprinted: 11150
Segment 1
handcuffs: 2729
handful: 42307
handicap: 67884
handiwork: 16912
handkerchief: 20952
handprinted: 15725
Segment 2
handful: 44662
handicap: 70836
handiwork: 45521
handlebars: 3869
handoff: 5741
handprinted: 33632
Segment 3
+
Compaction and merging process
handbag: 8786
handcuffs: 2729
handful: 44662
handicap: 70836
handiwork: 45521
handkerchief: 20952
handlebars: 3869
handoff: 5741
handprinted: 33632
Merged 1, 2, 3


Sparse index
in memory
key
byte offset
…
…
hammock
100491
handbag
102134
handsome
104667
hangout
106812
…
…
Sorted segment file (SSTable) on disk
……
hand: 91541
handbag: 8786
handcuffs: 2729
handful: 44662
handicap: 70836
handiwork: 45521
handkerchief: 20952
handlebars: 3869
handoff: 5741
handprinted: 33632
handsome: 86478
handwaving: 44005
handwriting: 22846
……
compressible block

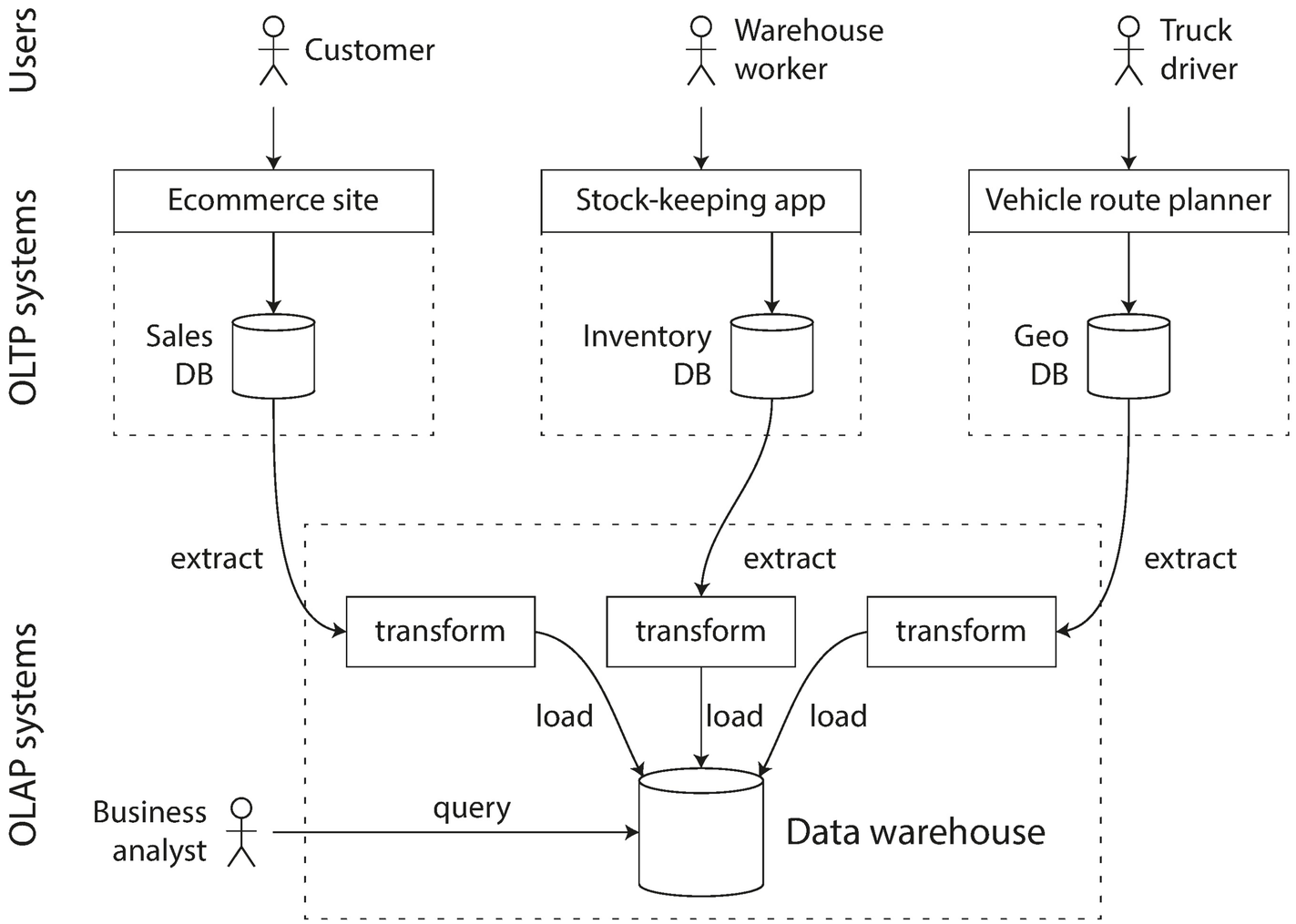
B-Tree


"Look up user_id = 251"
ref 100 ref 200 ref 300 ref 400 ref 500 ref
key < 100
100 ≤ key < 200
200 ≤ key < 300
key ≥ 500
400 ≤ key < 500
300 ≤ key < 400
ref 111 ref 135 ref 152 ref 169 ref 190 ref
ref 210 ref 230 ref 250 ref 270 ref 290 ref
250 ≤ key < 270
250 val 251 val 252 val 253 val 254 val

transaction processing versus analytic systems

Property	Transaction processing systems (OLTP)	Analytic systems (OLAP)
Main read pattern	Small number of records per query, fetched by key	Aggregate over large number of records
Main write pattern	Random-access, low-latency writes from user input	Bulk import (ETL) or event stream
Primarily used by	End user/customer, via web application	Internal analyst, for decision support
What data represents	Latest state of data (current point in time)	History of events that happened over time
Dataset size	Gigabytes to terabytes	Terabytes to petabytes

Data Warehouse



dim_product table

product_sk	sku	description	brand	category
30	OK4012	Bananas	Freshmax	Fresh fruit
31	KA9511	Fish food	Aquatech	Pet supplies
32	AB1234	Croissant	Dealicious	Bakery

dim_store table

store_sk	state	city
1	WA	Seattle
2	CA	San Francisco
3	CA	Palo Alto

fact_sales table

date_key	product_sk	store_sk	promotion_sk	customer_sk	quantity	net_price	discount_price
140102	31	3	NULL	NULL	1	2.49	2.49
140102	69	5	19	NULL	3	14.99	9.99
140102	74	3	23	191	1	4.49	3.89
140102	33	8	NULL	235	4	0.99	0.99

dim_date table

date_key	year	month	day	weekday	is_holiday
140101	2014	jan	1	wed	yes
140102	2014	jan	2	thu	no
140103	2014	jan	3	fri	no

dim_customer table

customer_sk	name	date_of_birth
190	Alice	1979-03-29
191	Bob	1961-09-02
192	Cecil	1991-12-13

dim_promotion table

promotion_sk	name	ad_type	coupon_type
18	New Year sale	Poster	NULL
19	Aquarium deal	Direct mail	Leaflet
20	Coffee & cake bundle	In-store sign	NULL