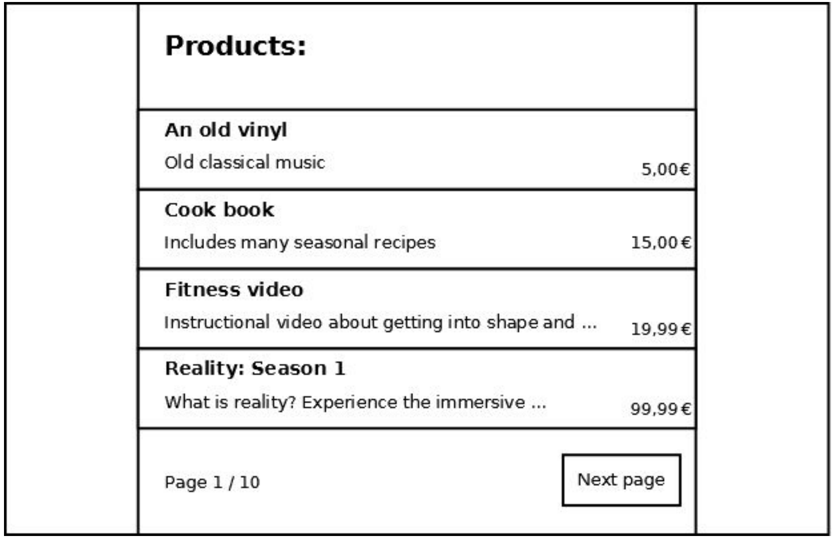
Distributed database systems
Project assignments

Juha Suomela
Arttu Tolvanen

Project 1: Problem description

- Multi-location, multi-database web shop application
- Sells collector’s items in Finland and northern Europe
- Client establishes warehouses in target countries
    - Could benefit greatly from a distributed database management system
        - Each site stores data that is not necessary at other sites (Storage, Orders)
        - Access to other sites still helpful (Login credentials, Cross-region purchases)
    - Similar to Amazon’s regional web shops, Amazon.com, Amazon.de, etc.

Project 1: Storyboards



Products:

An old vinyl

Old classical music 5,00€

Cook book

Includes many seasonal recipes 15,00€

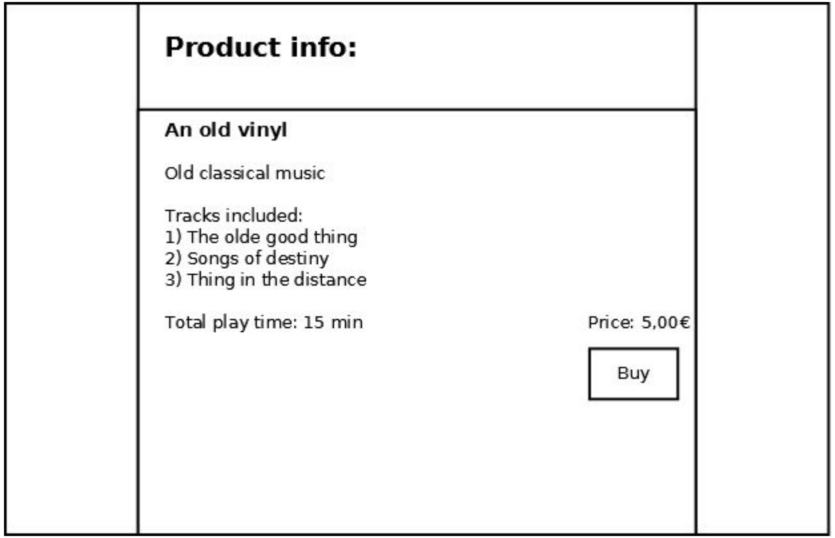
Fitness video

Instructional video about getting into shape and ... 19,99€

Reality: Season 1

What is reality? Experience the immersive ... 99,99€

Page 1 / 10



Product info:

An old vinyl

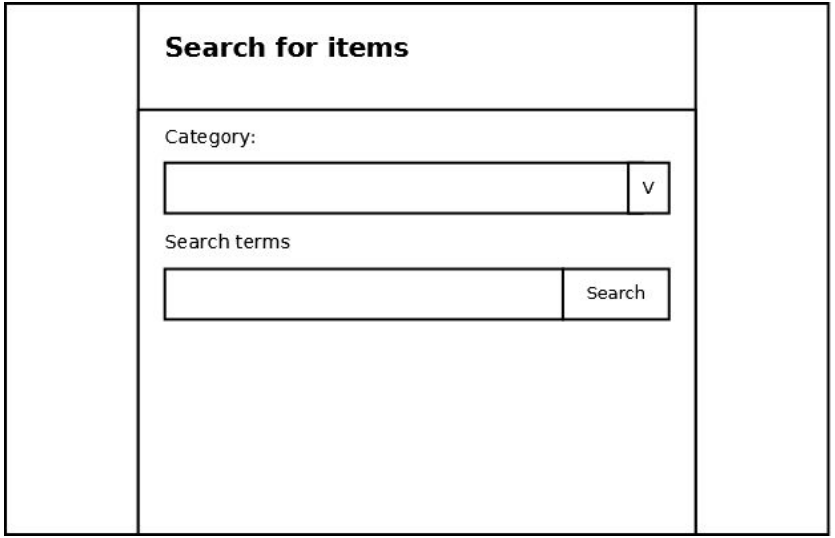
Old classical music

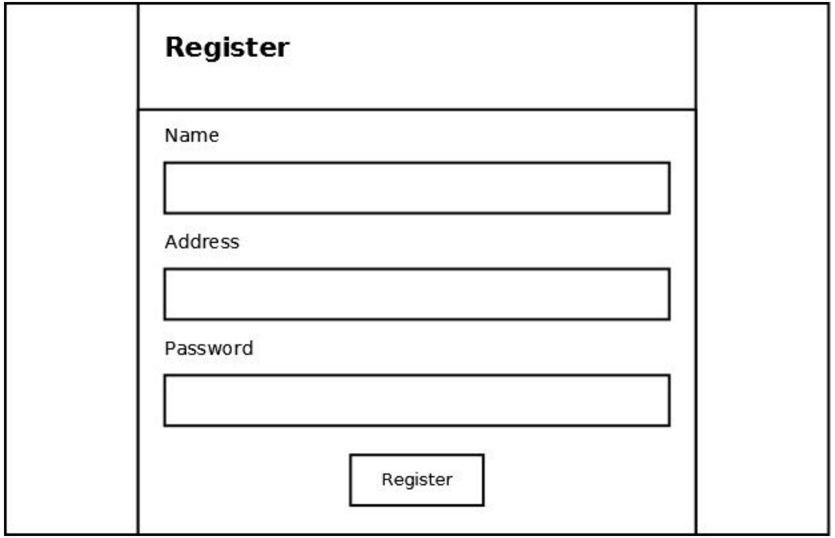
Tracks included:
1) The olde good thing
2) Songs of destiny
3) Thing in the distance

Total play time: 15 min

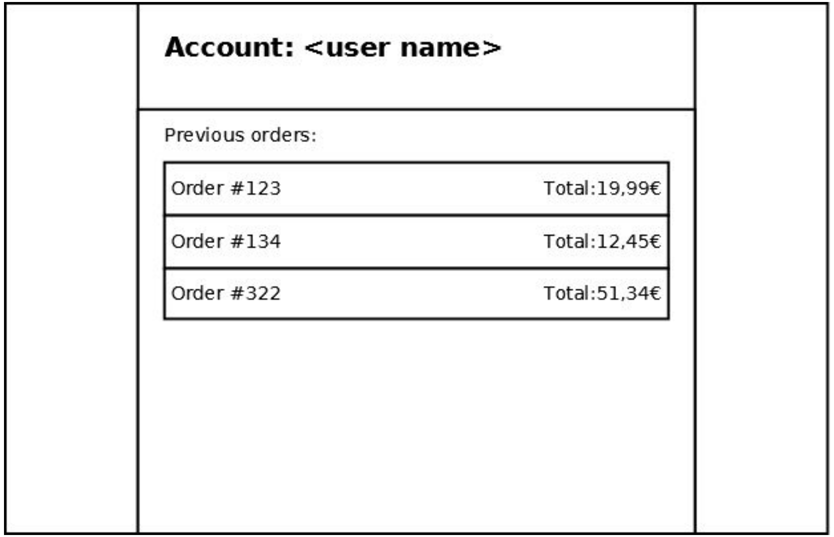
Price: 5,00€

Project 1: Storyboards





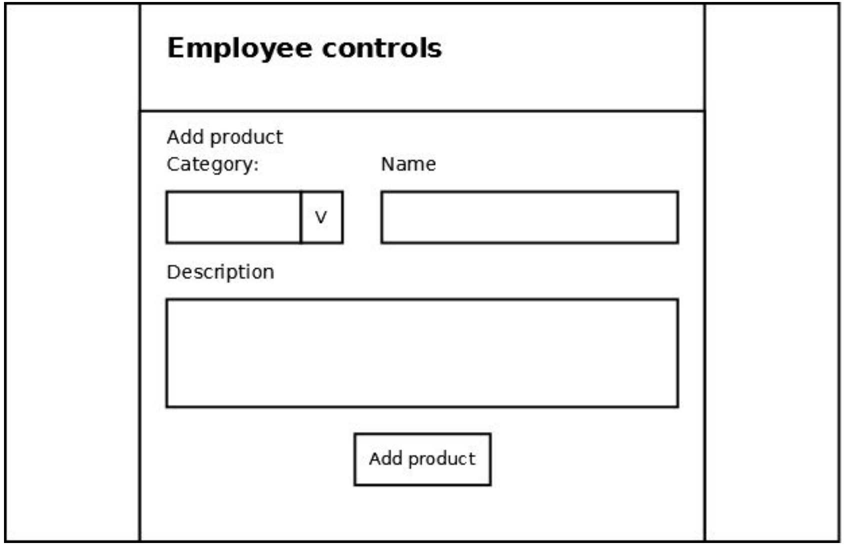
Project 1: Storyboards



Account: <user name>

Previous orders:

Order #123	Total:19,99€
Order #134	Total:12,45€
Order #322	Total:51,34€



Employee controls

Add product

Category: V

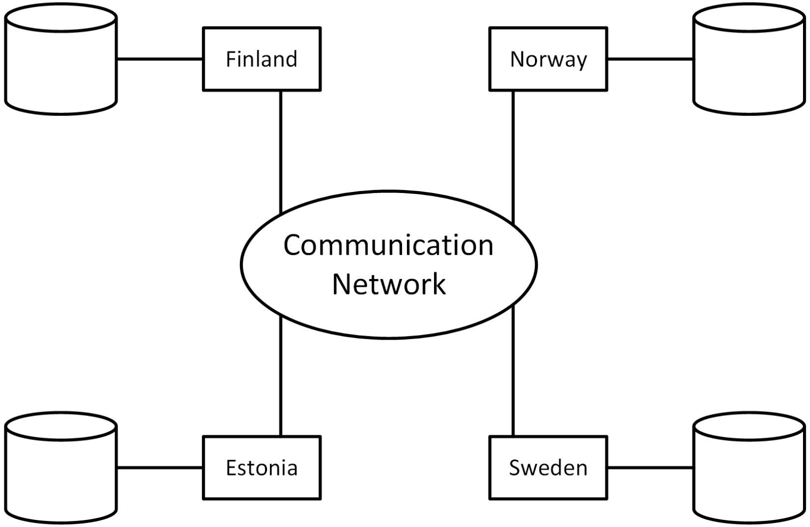
Name

Description

Project 1: Network model

Finland Orders, Finland OrderDetails, Finland Storage
Users, Items, Books, Vinyl, VHS

Norway Orders, Norway OrderDetails, Norway Storage
Users, Items, Books, Vinyl, VHS



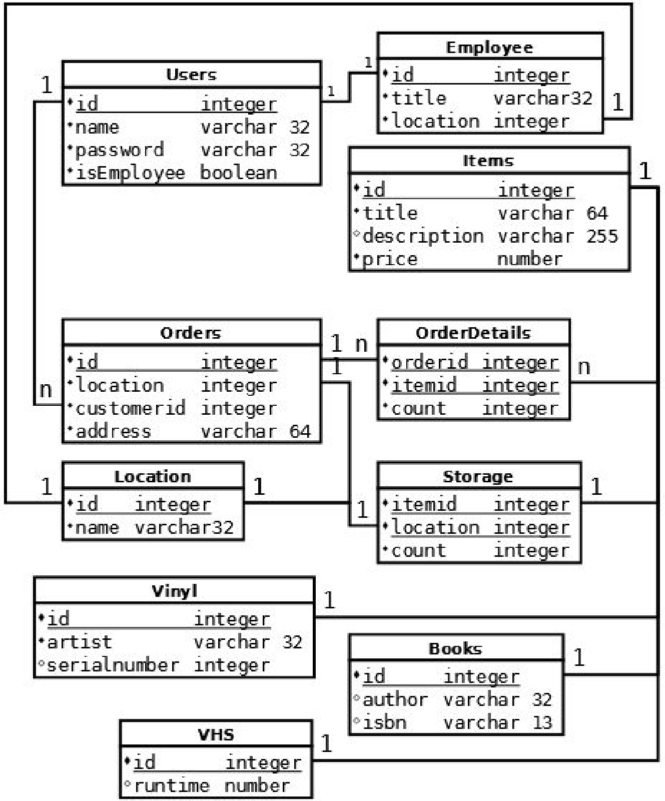
Estonia Orders, Estonia OrderDetails, Estonia Storage
Users, Items, Books, Vinyl, VHS

Sweden Orders, Sweden OrderDetails, Sweden Storage
Users, Items, Books, Vinyl, VHS

Project 1: Technical specification

- rqlite
  - A distributed database based on SQLite
  - Lightweight and free
- OpenVPN
  - To establish the communication network between sites
  - Free, secure, and reliable

Project 1: Schema



Users

◆	id	integer
◆	name	varchar 32
◆	password	varchar 32
◆	isEmployee	boolean

Employee

◆	id	integer
◆	title	varchar32
◆	location	integer

Items

◆	id	integer
◆	title	varchar 64
◇	description	varchar 255
◆	price	number

Orders

◆	id	integer
◆	location	integer
◆	customerid	integer
◆	address	varchar 64

OrderDetails

◆	orderid	integer
◆	itemid	integer
◆	count	integer

Location

◆	id	integer
◆	name	varchar32

Storage

◆	itemid	integer
◆	location	integer
◆	count	integer

Vinyl

◆	id	integer
◆	artist	varchar 32
◇	serialnumber	integer

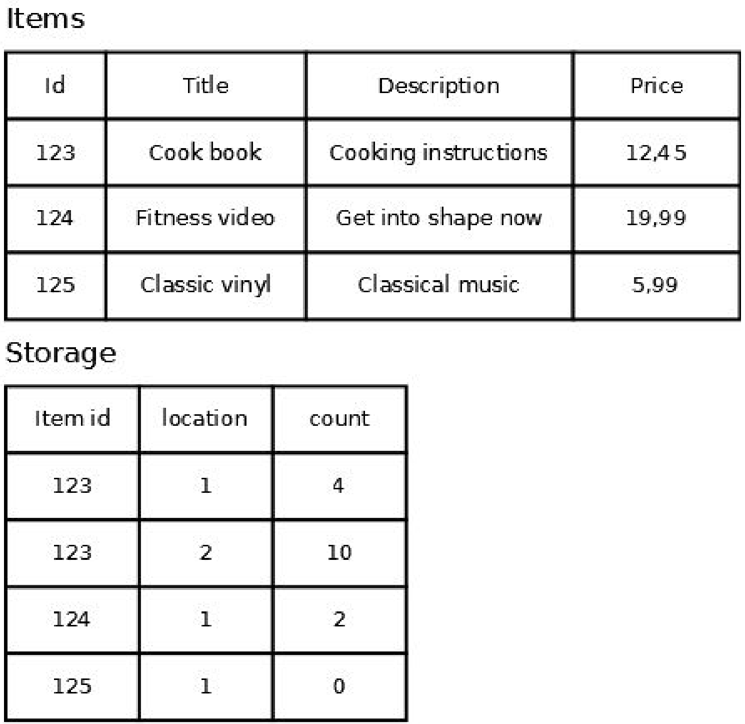
Books

◆	id	integer
◇	author	varchar 32
◇	isbn	varchar 13

VHS

◆	id	integer
◇	runtime	number

Project 1: Instances

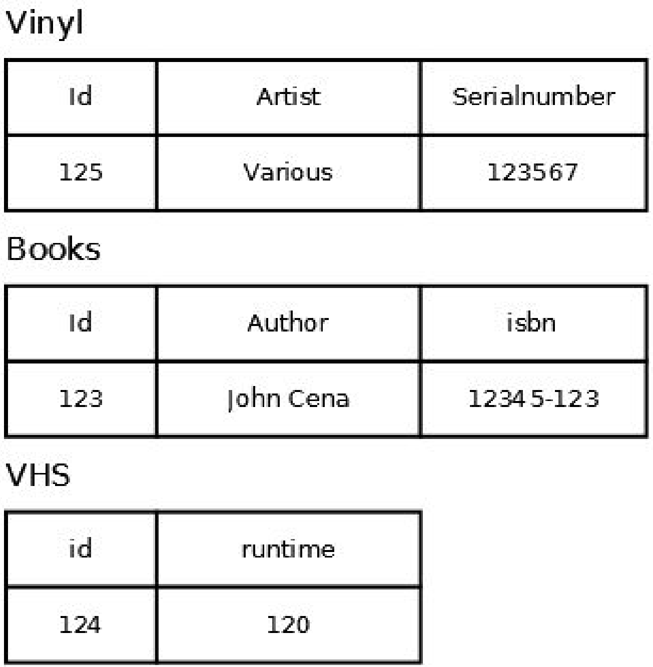


Items

Id	Title	Description	Price
123	Cook book	Cooking instructions	12,45
124	Fitness video	Get into shape now	19,99
125	Classic vinyl	Classical music	5,99

Storage

Item id	location	count
123	1	4
123	2	10
124	1	2
125	1	0



Vinyl

Id	Artist	Serialnumber
125	Various	123567

Books

Id	Author	isbn
123	John Cena	12345-123

VHS

id	runtime
124	120

Project 1: Fragmentation

- Horizontal fragmentation
  - Storage, Orders, OrderDetails
  - Only storing the necessary data at each site
- Vertical fragmentation
  - Items relation, according to ItemID and Price attributes
  - Hypothetical, helpful for accounting
- Do not fragment
  - Users, Employees, Items (incl. Books, VHS, Vinyl)
- Cost of fragmentation
  - Low transmission costs
  - Other costs scale with number of users

Project 1: Integration and access control

- On-Line Transaction Processing application
  - High volume of transactions
  - Requires up-to-date data
- Logical integration
  - Global conceptual schema is virtual
  - All data resides in operational databases
- Data and access control
  - Materialized views for neighboring site Storage fragments
  - Multilevel access control
  - Structural constraints to provide semantic integrity control

Project 1: Query processing

- Decomposition
  - Similar to a centralized database
- Localization
  - Viewing orders from different regions
  - Cross-region purchases
  - Primary horizontal fragmentation reduction
- Optimization
  - Total cost estimation
- Execution
  - Database homogeneity

Project 1: Transaction management

- Database consistency
  - Atomic, complete, isolated, and durable transactions
  - Query end result is a valid database even if errors occur
  - Provided by the DBMS

- Purchase transaction
  - Rollback to previous state if product in order missing from stock

Project 1: Concurrency control

- Locking
  - Editing data located in multiple sites
  - Managed by a centralized lock manager

- Deadlock avoidance
  - Consistent acquisition order of locks

Project 1: Reliability

- Transaction failures
  - Rollback to last consistent state
  - Handled by DBMS
- Physical failure
  - RAID-storage to increase fault tolerance
  - Backups for recovery
- Communication failure
  - Two-phase commit protocol

Project 1: Replication

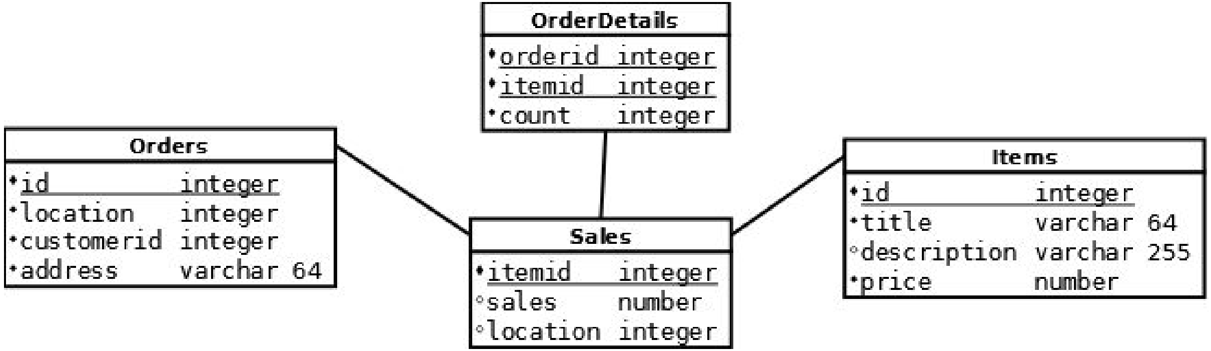
- Performance-focused strategy
  - Maximizing locality of reference
  - Only replicating the necessary data
  - Risk for data loss
- Fragmented relations
  - Orders and OrderDetails loss problematic due to customer returns
  - Storage loss could be recovered from with materialized views
- Backups
  - Eliminate the need to replicate for redundancy
  - Worst-case scenarios for data loss can be processed with a delay

Project 1: Data warehouse design

- Design goal
  - Provide the client with the ability to analyze product and site performance

- Implementation
  - Query the application database to build a separate data warehouse
  - Sums up all sales of each product for each location
  - Query logic
    - Multiply the ProductDetails.count by Items.price and multiply result together.
    - Group result together where Orders.id = OrderDetails.orderid.

Project 1: Star schema



OrderDetails

•orderid	integer
•itemid	integer
•count	integer

Orders

•id	integer
•location	integer
•customerid	integer
•address	varchar 64

Sales

•itemid	integer
◦sales	number
◦location	integer

Items

•id	integer
•title	varchar 64
◦description	varchar 255
•price	number

Project 2: Analyzing the DWH with Weka

- Weka environment explored in detail in the report
- Product recommendations
    - Extending the data warehouse schema to identify similar products
        - Adding subcategories to the product database (for example, genre)
    - Identifying products that are purchased together
- Data mining tools
    - OneR classifier
    - J48 decision tree classifier
    - NaiveBayes classifier

Project 2: Training data

(accumulated reviews of all users)

```
@relation shop

@attribute item numeric
@attribute genre {scifi, romance, action}
@attribute review {good, bad}

@data
1, scifi, good
1, scifi, good
1, scifi, good
1, scifi, bad
2, romance, good
2, romance, bad
2, romance, good
3, romance, good
3, romance, good

....
```

Project 2: Test data

(single users review scores)

```
@data
1, scifi, good
6, scifi, good
```

Project 2: Output

Results from: NaiveBayes, J48 and OneR

```
Correctly Classified Instances      2           100     %
Incorrectly Classified Instances    0             0     %
```

- Clearly shows that user likes same kind of shows that most other users do
- If test data reviews would be “bad”, “Incorrectly Classified Instances” would show that user dislikes the types of shows that most other people do

Project 2: Summary

- We had some ideas, but fell short on understanding how to implement them
  - Apriori associator to identify products that are often bought together
  - OneR classifier to identify poorly performing products
  - Clustering customers based on purchasing patterns

- Generating a meaningful data set on our own proved difficult