SELECTING AN EFFECTIVE DATABASE

Thomas Swed

Lewis University

CPSC – 59700 Research Project

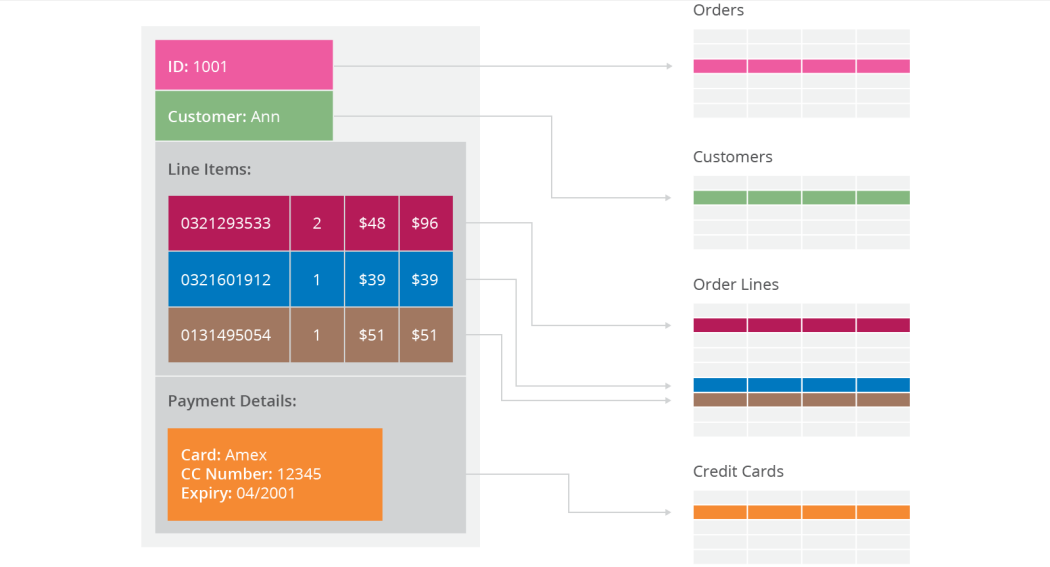
INTRODUCTION

- Types of databases
  - Relational – mature, tested
  - NoSQL – new, distributed
- Rise of NoSQL
  - Explosion of data generation from web
- How to choose?

RELATIONAL DATABASES

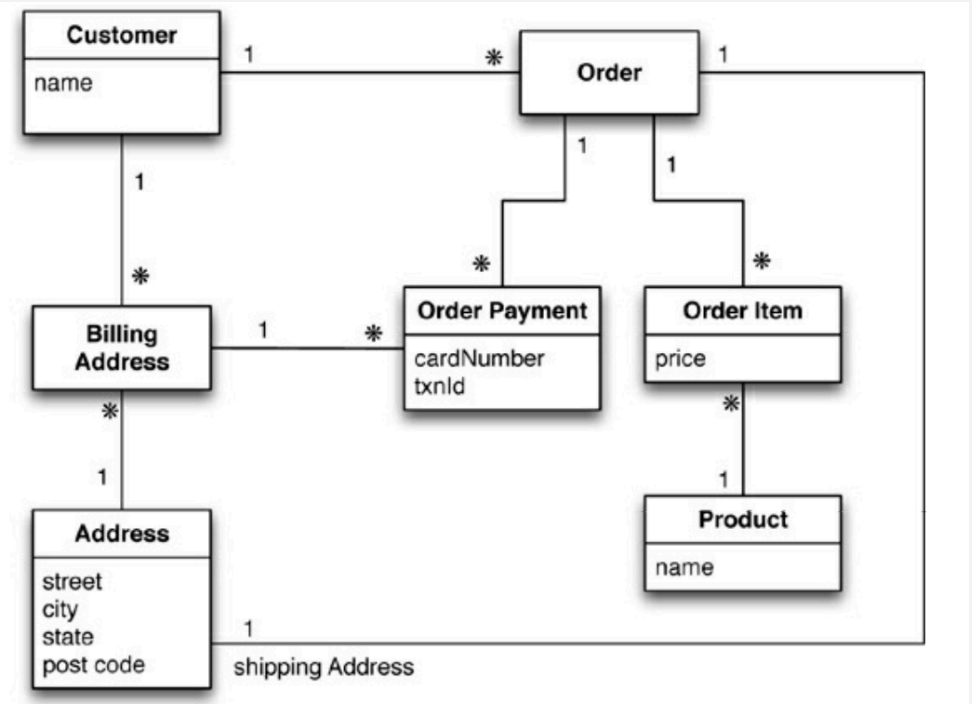
CHARACTERISTICS

- Data storage
  - Tables (Rows and columns)
- Relationships – Foreign keys
- Query language
  - SQL varies, core remains the same
- Schema
  - Types must be defined



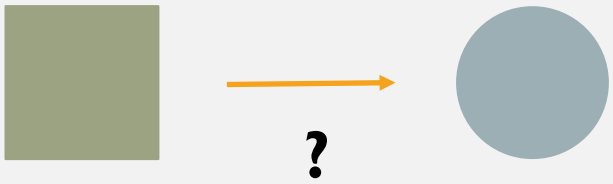
STRENGTHS

- ACID Transactions
  - Atomicity, consistency, isolation, durability
- Concurrency
- Simple
  - Tables
- Normalize Data
  - Updates



CHALLENGES

- Impedance Mismatch
- Costs
  - Hardware
  - Software
- Structure
  - Rise of unstructured data from websites
- Size
  - Limited amount of data



THE RISE OF NOSQL

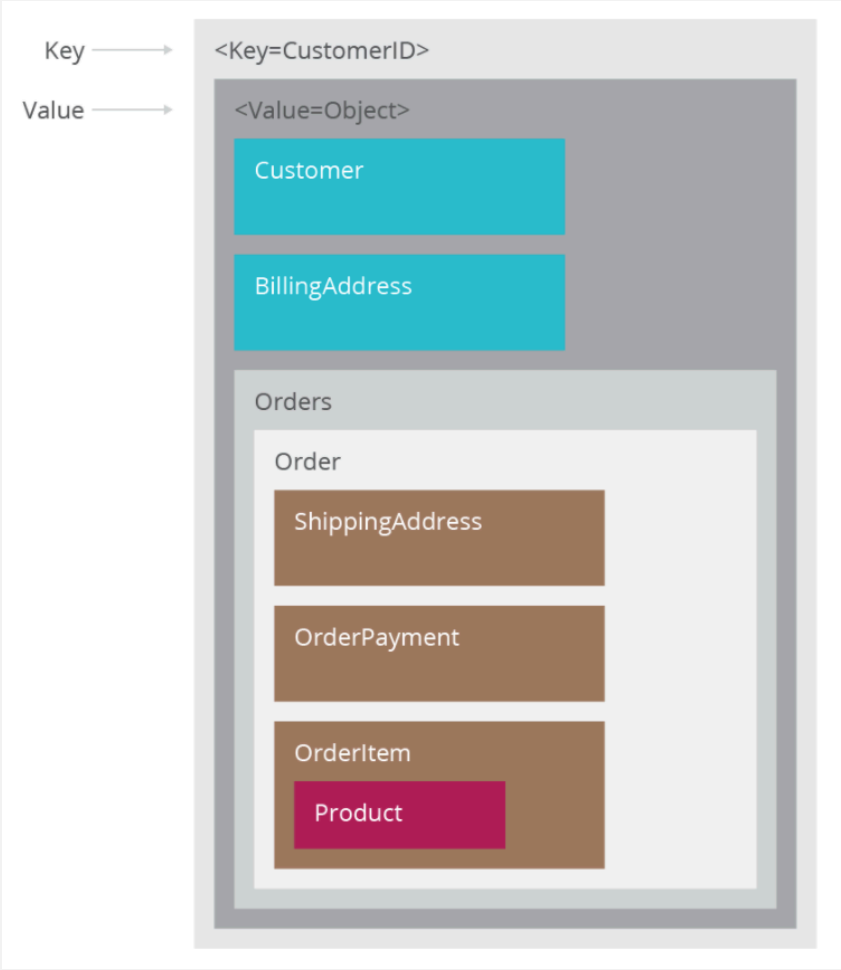
DIGITAL AGE

- Data generation
  - Big Data
- Hardware advances
  - Lower costs
- Scaling out as opposed to up

TYPES OF NOSQL DATABASES

KEY-VALUE

- Map/Dictionary structure
- When to use

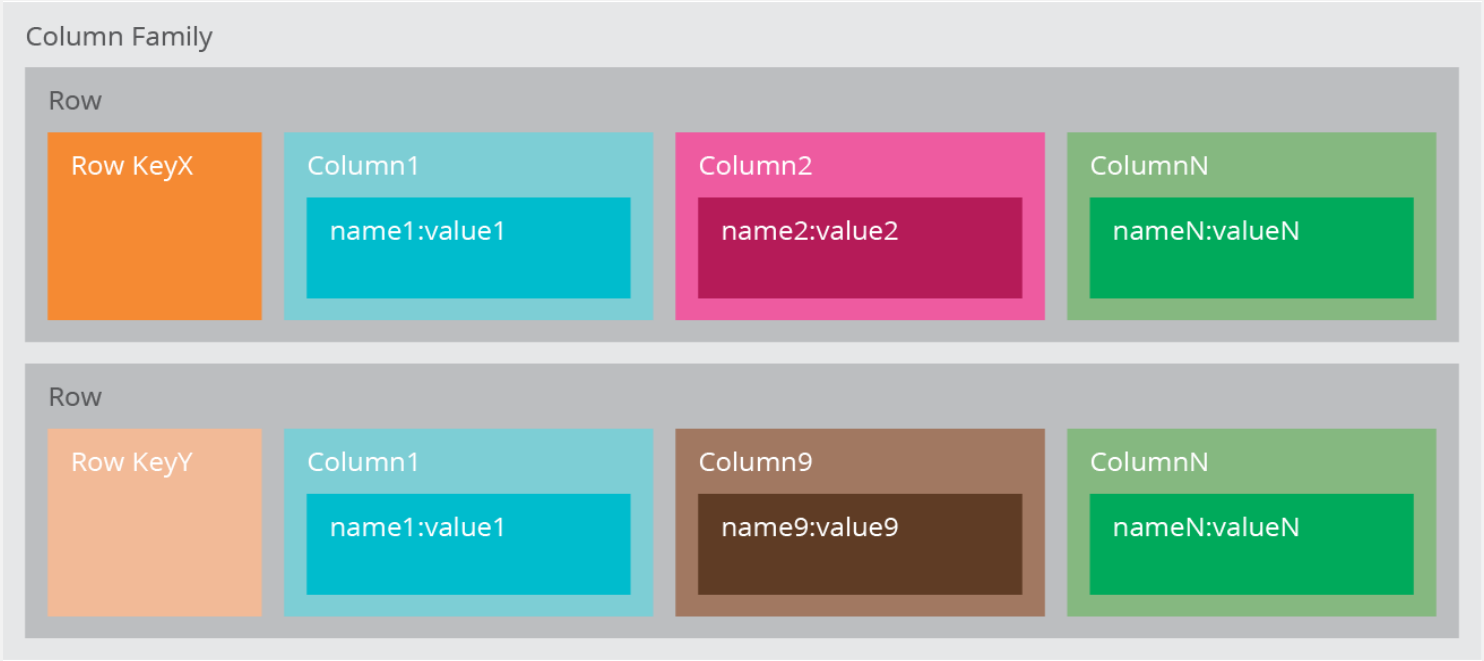


DOCUMENT

- Stored as a single record
- JSON, XML, YAML and BSON
- When to use

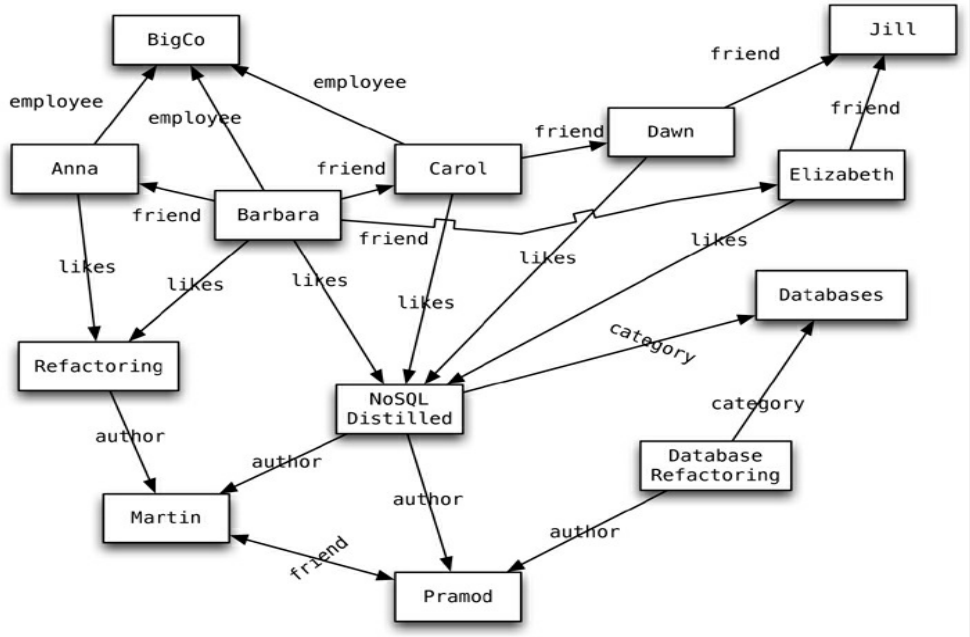
COLUMN-FAMILY

- Similar to key-value
- When to use



GRAPH

- Highly relational
- Designed for complex relationships
- When to use



DATA MODELS

WHAT IS A DATA MODEL

- A collection of conceptual tools used to model representations of real-world entities and the relationships among them

TYPES

RELATIONAL

- Data stored in tables
- Simple
- Traditional database
- Normalized

AGGREGATE

- Data stored in records or documents
- Allows for complex structure
- New and open source
- Denormalized

DATA DISTRIBUTION

REPLICATION

- Duplicates across nodes
- Two forms
  - Master-slave
  - Peer-to-peer

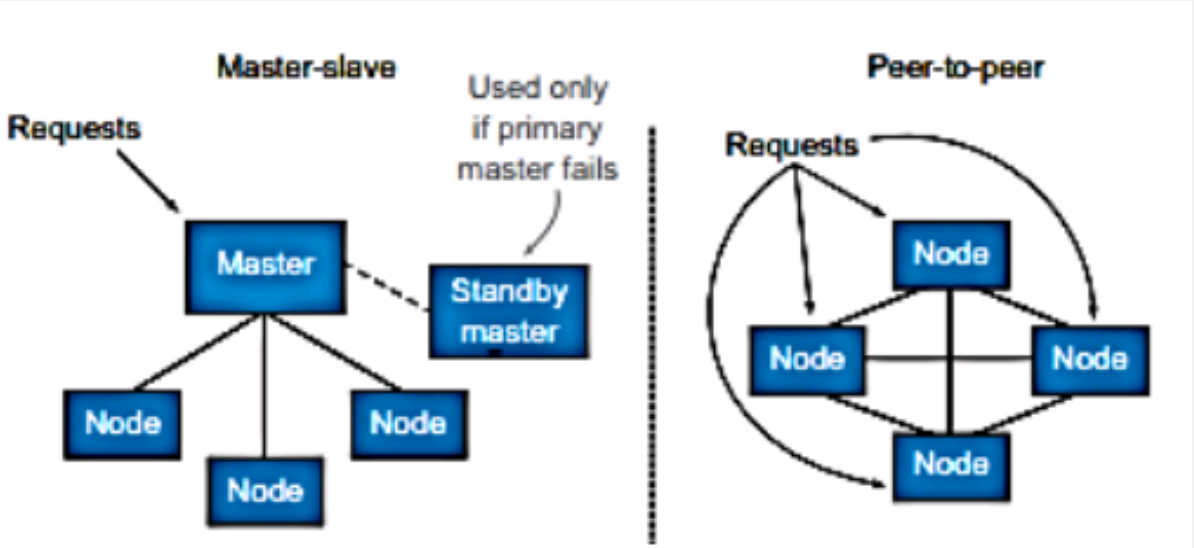
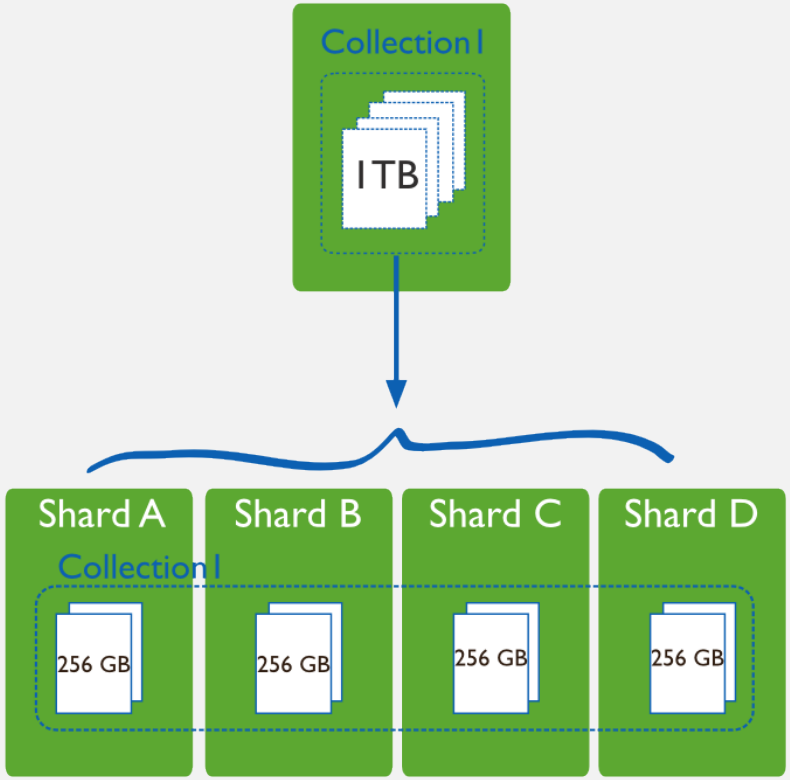


Figure: Master-slave versus peer-to-peer

SHARDING

- Distributes data across nodes
- Increases performance
- Easily implemented with aggregate models



CHOOSING A DATABASE

RELATIONAL

- Manages concurrency well through ACID transactions
- Mature
- Can handle queries against multiple transactions

NOSQL

- Programmer productivity
  - Impedance mismatch
- Data-access performance
  - Scale to accommodate huge data sets
  - Performance

CONCLUSION

- Different databases serve different needs
- Discern which type works best for a given problem
- Use a conglomerate of types