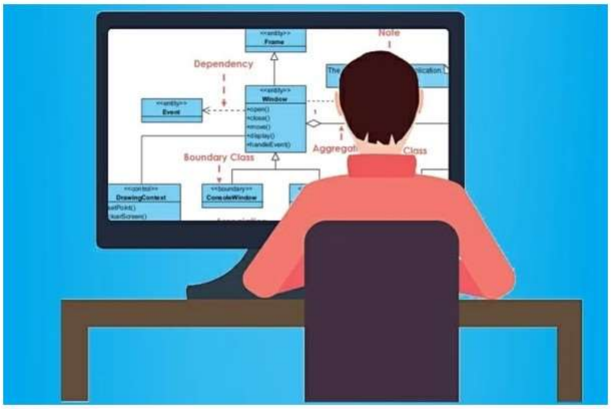
Software Design & Architecture
Spring 2022 - Week-05



مدرس: مہندس ماجد کلیم

جامعہ بحریہ، واقعگاہ کراچی

Engr. Majid Kaleem

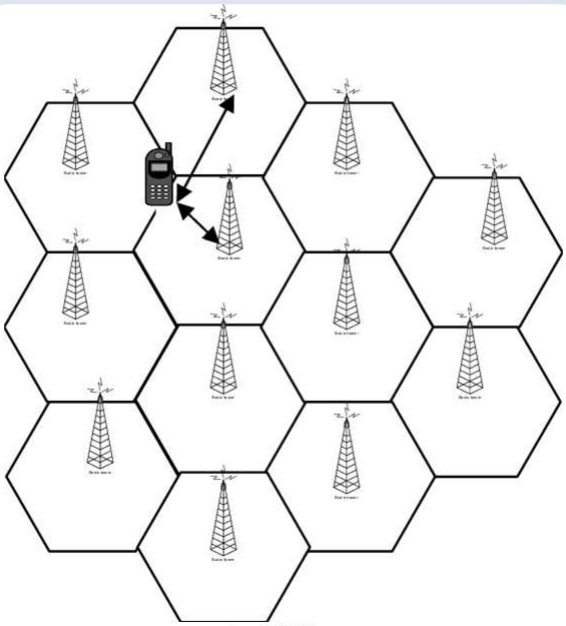
WEEKLY AGENDA

TENTATIVE WEEKLY DATES		TENTATIVE TOPICS
1	Mar 7th – Mar 11th	INTRODUCTION TO THE COURSE; DEFINING SOFTWARE ARCHITECTURE & DESIGN CONCEPTS
2	Mar 14th – Mar 18th	DESIGN PRINCIPLES; OBJECT-ORIENTED DESIGN WITH UML
3	Mar 21st – Mar 25th	SYSTEM DESIGN & SOFTWARE ARCHITECTURE; OBJECT DESIGN, MAPPING DESIGN TO CODE
4	Mar 28th – Apr 1st	FUNCTIONAL DESIGN; UI DESIGN; WEB APPLICATIONS DESIGN **ASSIGNMENT & QUIZ #1**
5	Apr 4th – Apr 8th	MOBILE APPLICATION DESIGN; PERSISTENCE LAYER DESIGN
6	Apr 11th – Apr 15th	CREATIONAL DESIGN PATTERNS
7	Apr 18th – Apr 22nd	STRUCTURAL DESIGN PATTERNS **ASSIGNMENT & QUIZ #2**
8	Apr 25th – Apr 29th	BEHAVIORAL DESIGN PATTERNS
← MID TERM EXAMINATIONS →		
9	May 9th – May 13th	INTERACTIVE SYSTEMS WITH MVC ARCHITECTURE; SOFTWARE REUSE
10	May 16th – May 20th	ARCHITECTURAL DESIGN ISSUES; ARCHITECTURE DESCRIPTION LANGUAGES (ADLS)
11	May 23rd – May 27th	ARCHITECTURAL STYLES/PATTERNS & DESIGN QUALITIES
12	May 30th – Jun 3rd	ARCHITECTURAL STYLES/PATTERNS & DESIGN QUALITIES **ASSIGNMENT & QUIZ #3**
13	Jun 6th – Jun 10th	QUALITY TACTICS; ARCHITECTURE DOCUMENTATION
14	Jun 13th – Jun 17th	ARCHITECTURAL EVALUATION TECHNIQUES
15	Jun 20th – Jun 24th	MODEL DRIVEN DEVELOPMENT **ASSIGNMENT (PRESENTATIONS) & QUIZ #4**
16	Jun 27th – Jul 1st	REVISION WEEK
← FINAL TERM EXAMINATIONS →		

MOBILE APPLICATION DESIGN & ARCHITECTURE

- A dedicated course by the name: *"Software Applications for Mobile Devices"* is available in your roadmap to acquire in-depth knowledge on this topic.

MOBILE APPLICATION DESIGN & ARCHITECTURE



MOBILE APPLICATION DESIGN & ARCHITECTURE

- MobileApp design encompasses technical and nontechnical activities that include:
  - establishing the look and feel of the mobile application,
  - creating the aesthetic layout of the user interface,
  - establishing the rhythm of user interaction,
  - defining the overall architectural structure,
  - developing the content and functionality that reside within the architecture,
  - and planning the navigation that occurs within the MobileApp.
- Special attention needs to be given to the elements that add *context awareness* to the MobileApp
- https://whatis.techtarget.com/definition/context-awareness

MOBILE APPLICATION DESIGN - STEPS

- MobileApp design is similar to WebApp design and encompasses *six major steps* that are driven by information obtained during requirements modeling.
  1. Content design addresses the same issues for both WebApp and MobileApp design.
  2. During architectural design, MobileApp developers determine which functions will be implemented in the native app running on the mobile device and which will be implemented as Web or cloud services.
  3. Interface design establishes the layout and interaction mechanisms that define the user experience.

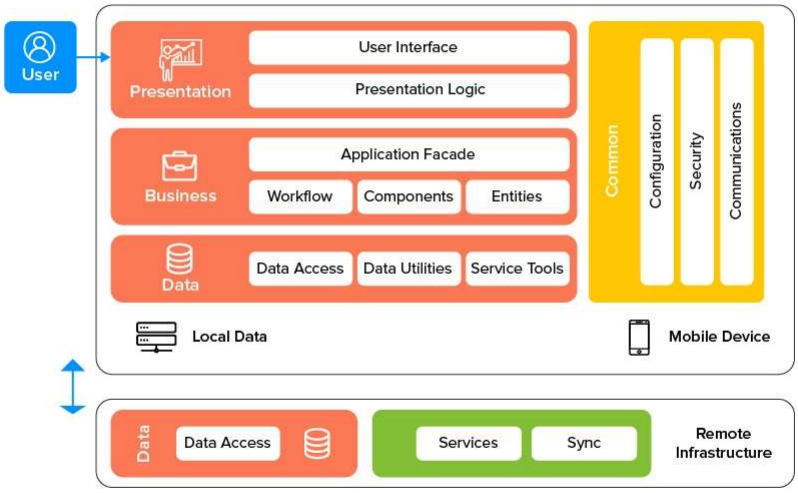
MOBILE APPLICATION DESIGN - STEPS

4. Ensuring that the MobileApp makes appropriate use of context affects both interface design and content design.
5. Navigation design defines how the end user navigates through the content structure.
6. Component design represents the detailed internal structure of functional elements of the MobileApp.

MOBILE APPLICATION DESIGN & ARCHITECTURE

- Mobile apps are typically structured using multilayered architectures, including a user interface layer, a business layer, and a data layer.
- With mobile apps you have the choice of building a thin Web-based client or a rich client.
- With a thin client, only the user interface resides on the mobile device, whereas the business and data layers reside on a server.
- With a rich client all three layers may reside on the mobile device itself.
- Mobile devices differ from one another in terms of their physical characteristics (e.g., screen sizes, input devices), software (e.g., operating systems, language support), and hardware (e.g., memory, network connections).

MOBILE APPLICATION DESIGN & ARCHITECTURE



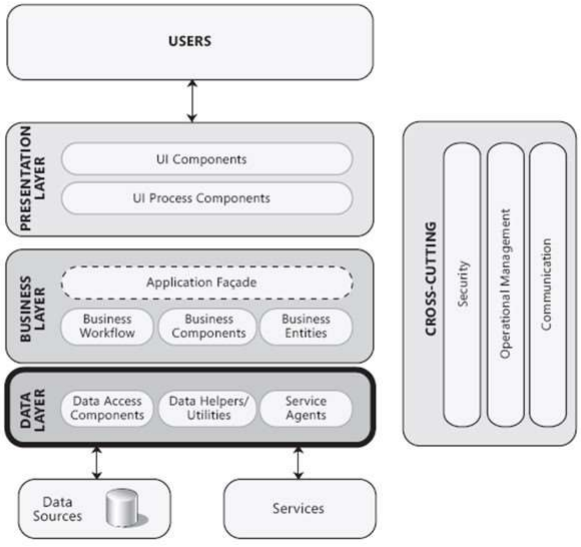
MOBILE APPLICATION DESIGN-USEFUL RESOURCES

- https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/
- https://os-system.com/blog/mobile-app-architecture-how-to-design-it/

PERSISTENCE LAYER DESIGN

- A dedicated course by the name: *"Relational Database Management Systems"* is available in your roadmap to acquire in-depth knowledge of persistence layer.
- Persistence layer design means designing of *database* or data store layer.
- Several styles are available as described on the following slides.

PERSISTENCE LAYER TYPICAL ARCHITECTURE



A typical application showing the data layer and the components it may contain

RELATIONAL DATABASE

Relation=Table

Relation (Student)

Std_ID	FName	LName	...	...
	------Related Data------			

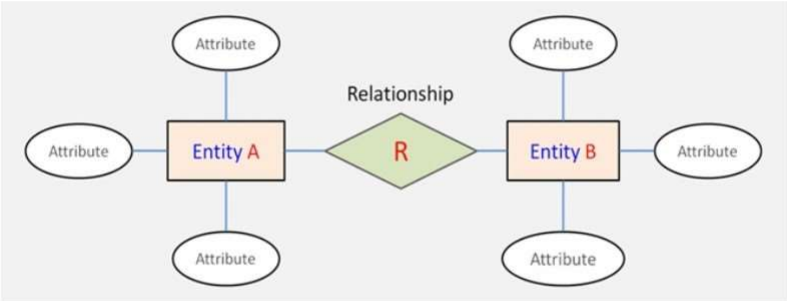
←*Relationship*

Relation (Courses)

Crs_ID	Std_ID	CName	CHrs	...
	------Related Data------			

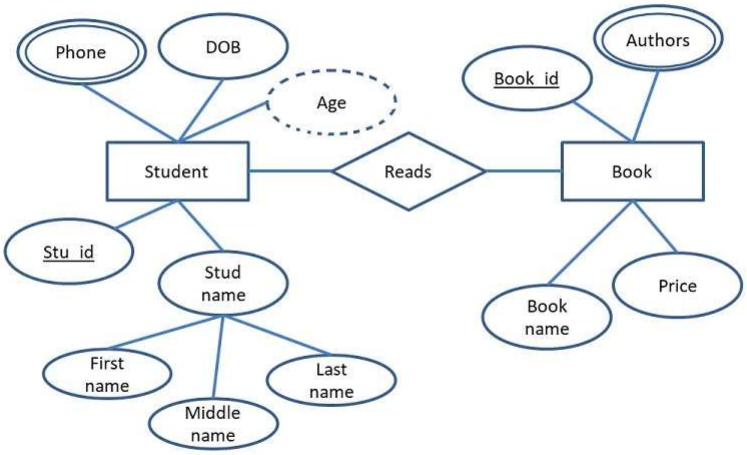
RELATIONAL DATABASE EXAMPLE

- Entity-Relationship Diagram (ERD) - Pictorial representation of a database model.



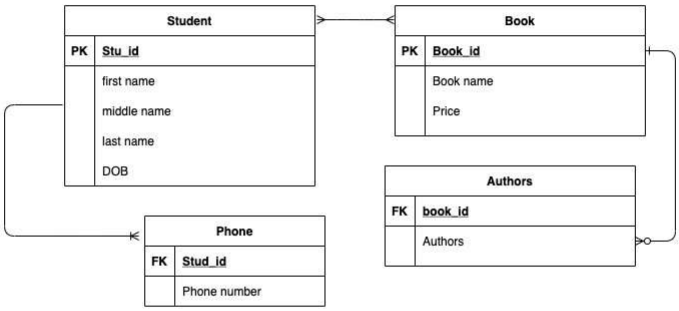
RELATIONAL DATABASE EXAMPLE

- ER Model:



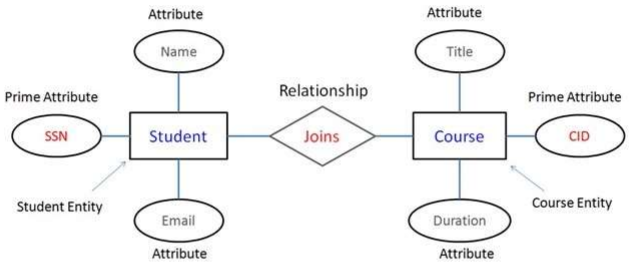
RELATIONAL DATABASE EXAMPLE

- Relational Model:



RELATIONAL DATABASE EXAMPLE

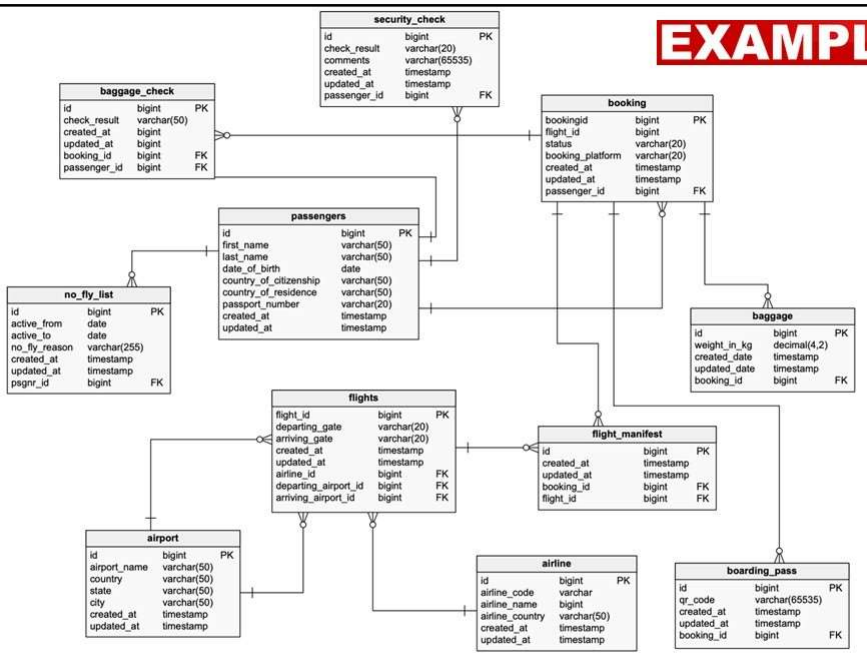
- Converting an ER model into Relational model:



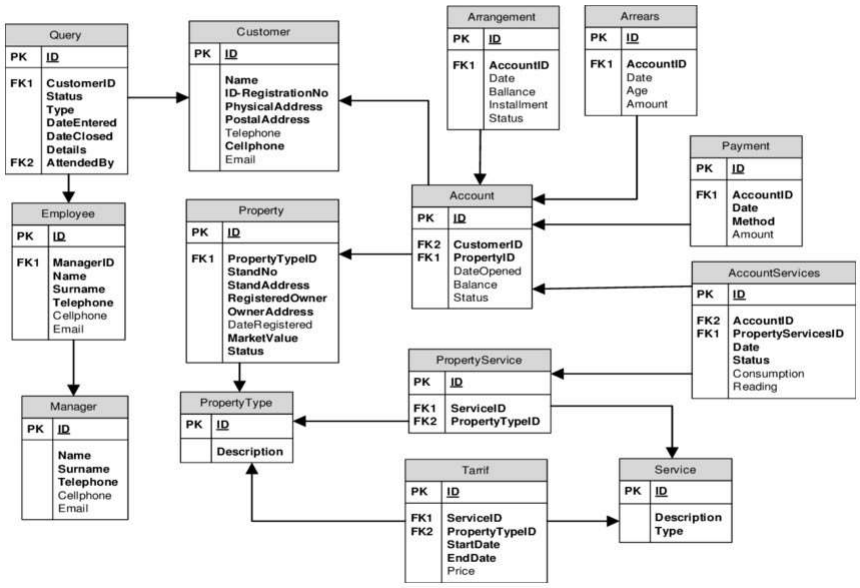
SSN	Name	Email

CID	Title	Duration

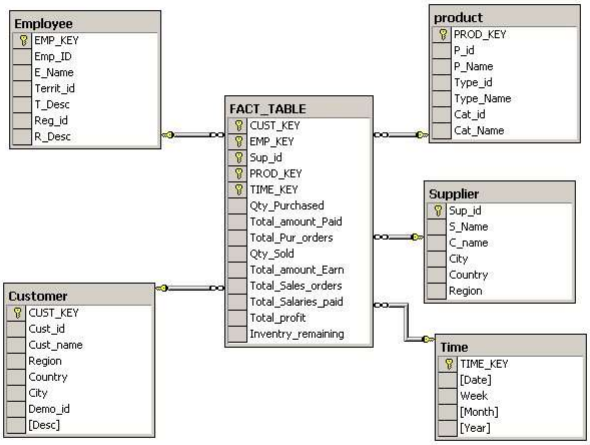
EXAMPLE



RELATIONAL DATABASE EXAMPLE

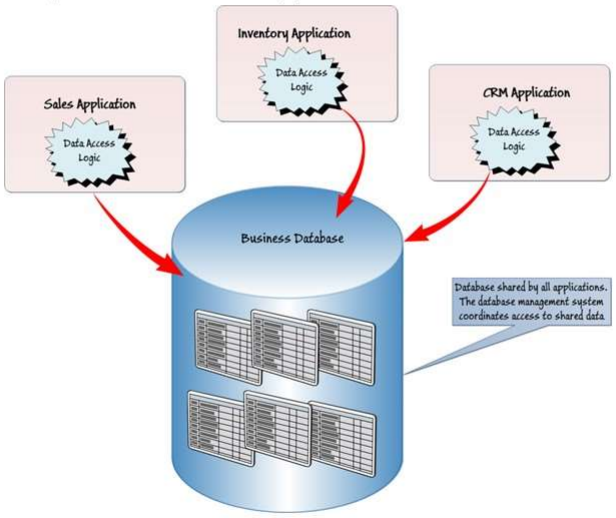


RELATIONAL DATABASE EXAMPLE



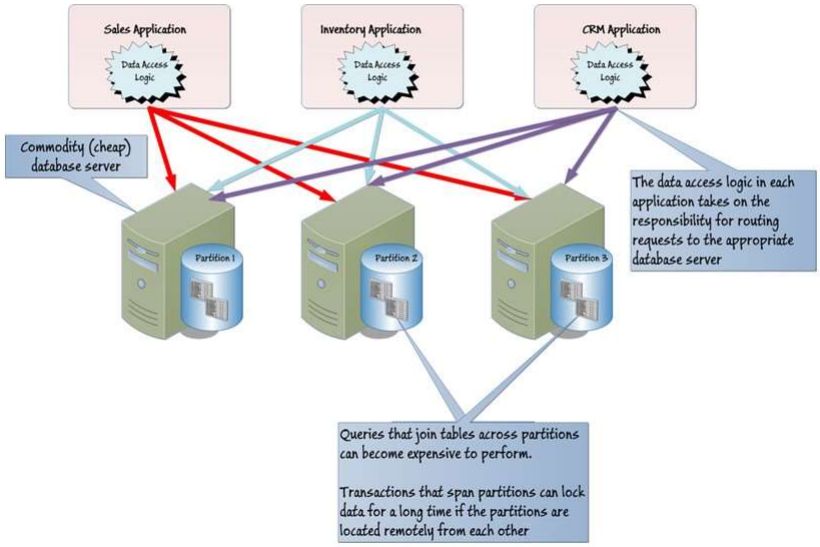
RELATIONAL DATABASE EXAMPLE

- Relational Databases where the database it designed as the integration point for business applications.



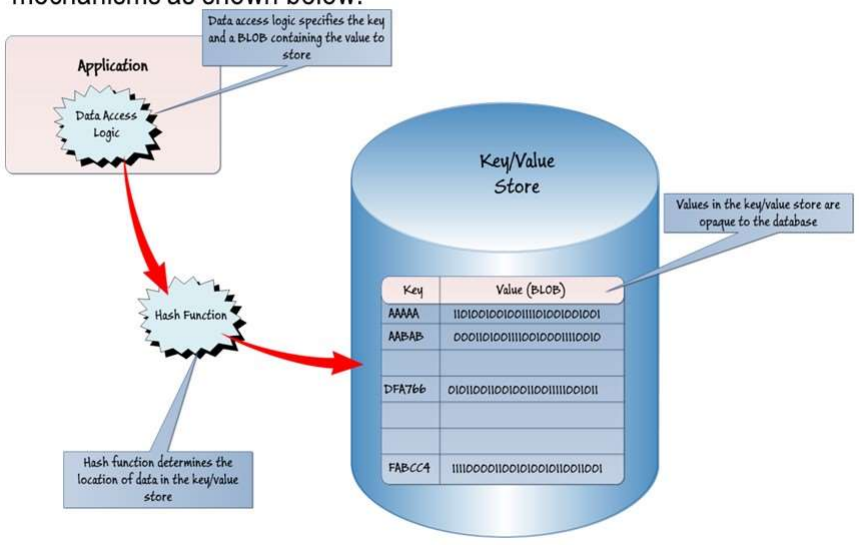
RELATIONAL DATABASE EXAMPLE

- Partitioning is used to scale out a database as shown below:



KEY/VALUE DATABASE EXAMPLE

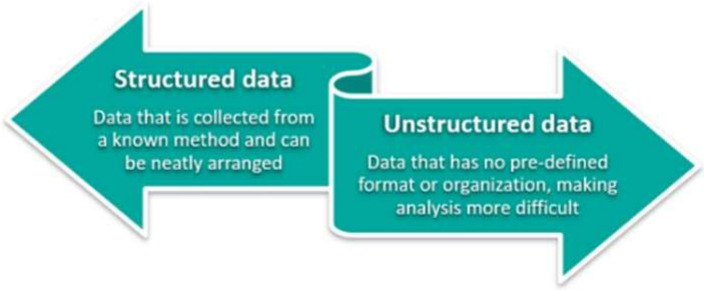
- A key/value store implements arguably the simplest of the NoSQL storage mechanisms as shown below:



23

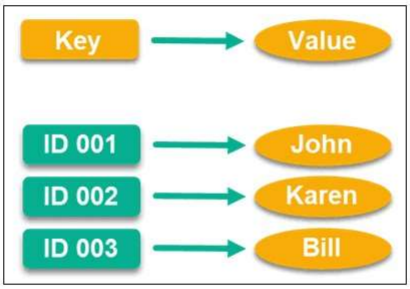
NoSQL

- Relational databases use “Structured Data”.
- NoSQL databases use “Structured Data/Unstructured Data”.
- NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables.
- They provide flexible schemas and scale easily with large amounts of data and high user loads.



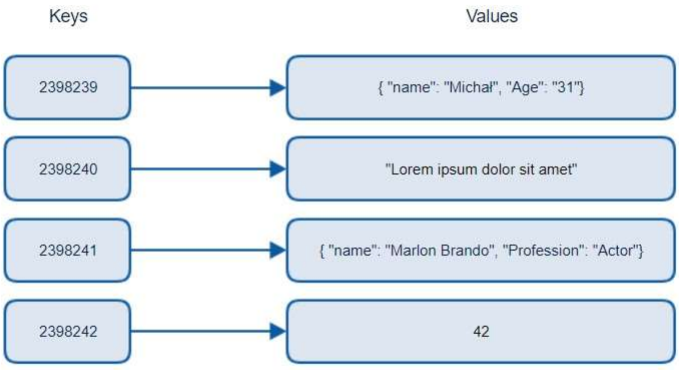
KEY/VALUE DATABASE EXAMPLE

- A key/value store implements arguably the simplest of the NoSQL storage mechanisms as shown below:



25

KEY/VALUE DATABASE EXAMPLE

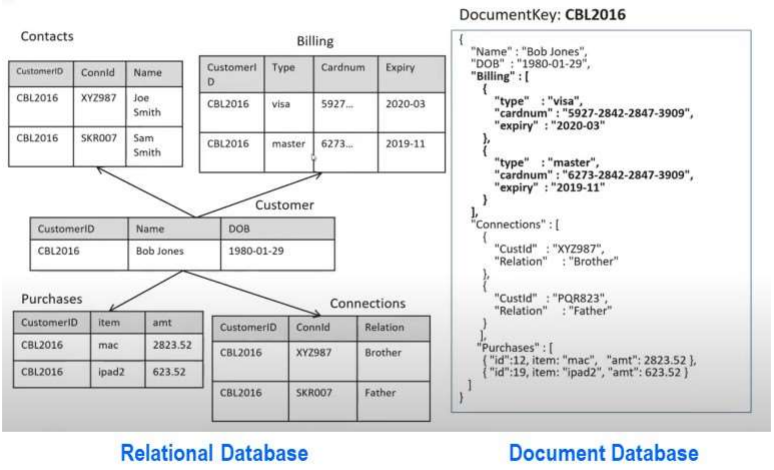


DOCUMENT DATABASE

- A *document database* is similar in concept to a key/value store except that the values stored are documents.
- A document is a collection of named fields and values, each of which could be simple scalar items or compound elements such as lists and child documents.
- The fields in the documents are exposed to the database management system, enabling an application to query and filter data by using the values in these fields.

27

DOCUMENT DATABASE EXAMPLE



DOCUMENT DATABASE EXAMPLE

Relational

ID	first_name	last_name	cell	city	year_of_birth	location_x	location_y
1	'Mary'	'Jones'	'516-555-2048'	'Long Island'	1986	'-73.9876'	'40.7574'

ID	user_id	profession
10	1	'Developer'
11	1	'Engineer'

ID	user_id	name	version
20	1	'MyApp'	1.0.4
21	1	'DocFinder'	2.5.7

ID	user_id	make	year
30	1	'Bentley'	1973
31	1	'Rolls Royce'	1965

MongoDB

```
{
first_name: "Mary",
last_name: "Jones",
cell: "516-555-2048",
city: "Long Island",
year_of_birth: 1986,
location: {
        type: "Point",
        coordinates: [-73.9876, 40.7574]
},
profession: ["Developer", "Engineer"],
apps: [
 { name: "MyApp",
   version: 1.0.4 },
 { name: "DocFinder",
   version: 2.5.7 }
],
cars: [
   { make: "Bentley",
     year: 1973 },
   { make: "Rolls Royce",
     year: 1965 }
 ]
}
```

Relational Database **Document Database**

DOCUMENT DATABASE EXAMPLE

Document 1

```
{
 "id": "1",
 "name": "John Smith",
 "isActive": true,
 "dob": "1964-30-08"
}
```

Document 2

```
{
 "id": "2",
 "fullName": "Sarah Jones",
 "isActive": false,
 "dob": "2002-02-18"
}
```

Document 3

```
{
 "id": "3",
 "fullName":
 {
   "first": "Adam",
   "last": "Stark"
 },
 "isActive": true,
 "dob": "2015-04-19"
}
```

DOCUMENT DATABASE EXAMPLE

Row Key	Document
1001	OrderDate: 06/06/2013 OrderItems: ProductID: 2010 Quantity: 2 Cost: 520 ProductID: 4365 Quantity: 1 Cost: 18 OrderTotal: 1058 Customer ID: 99 ShippingAddress: StreetAddress: 205 108th Ave City: Bellevue State: WA ZipCode: 98004
1002	OrderDate: 07/07/2013 OrderItems: ProductID: 1285 Quantity: 1 Cost: 120 OrderTotal: 120 Customer ID: 220 ShippingAddress: StreetAddress: 401 W . Front St City: Boise State: ID ZipCode: 83702

31

WIDE-COLUMN DATABASE EXAMPLE

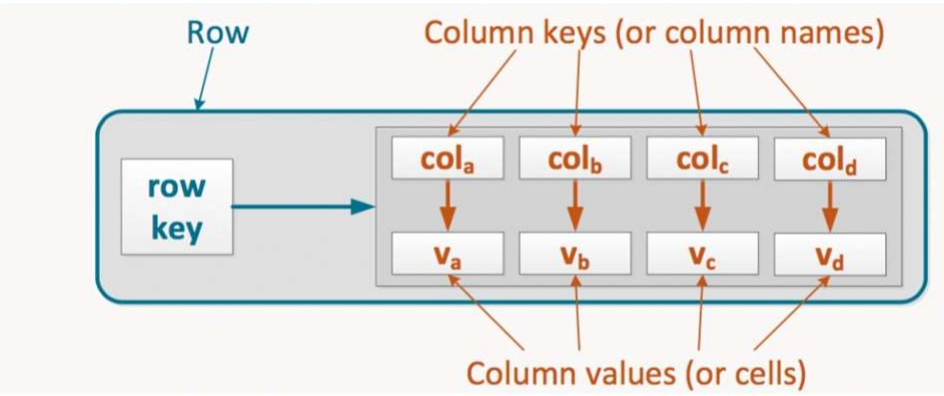
Row Store

Last Name	First Name	E-mail	Phone #	Street Address

Column Store

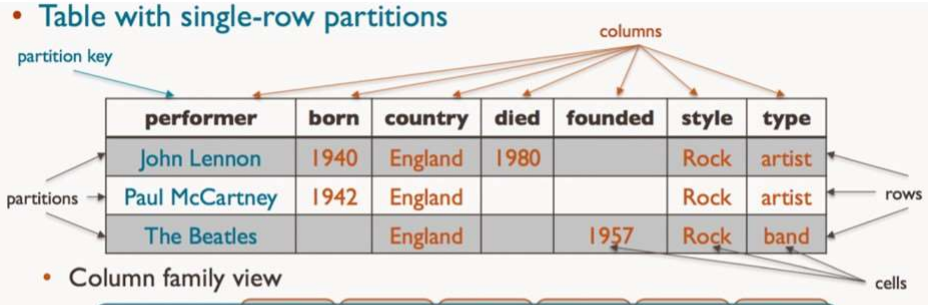
Last Name	First Name	E-mail	Phone #	Street Address

WIDE-COLUMN DATABASE EXAMPLE

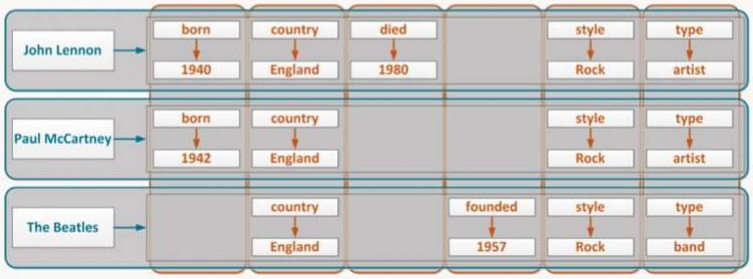


WIDE-COLUMN DATABASE EXAMPLE

- Table with single-row partitions



- Column family view

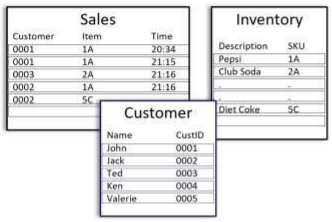


GRAPH DATABASE

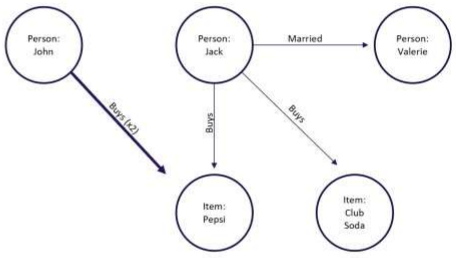
- Just like the other categories of *NoSQL* databases, a *graph database* enables you to store entities, but the main focus is on the relationships that these entities have with each other.
- A graph database stores two types of information; *nodes* which you can think of as instances of entities, and *edges* which specify the relationships between nodes.
- Nodes and edges can both have properties which provide information about that node or edge (like columns in a table).
- Additionally, edges can have a direction indicating the nature of the relationship as shown on the next slide.

35

GRAPH DATABASE EXAMPLE

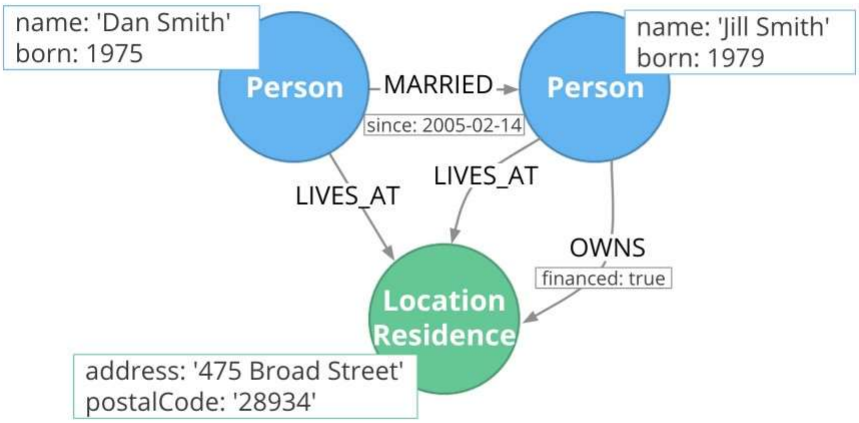


Traditional database store data to efficiently store facts, but relationships must be rebuilt with JOINs and other inexact techniques.

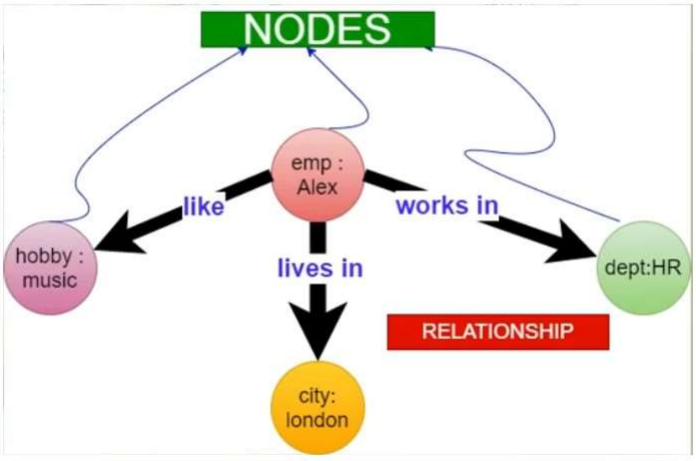


Graph databases store both facts and the relationships between the facts, making certain types of analysis more intuitive.

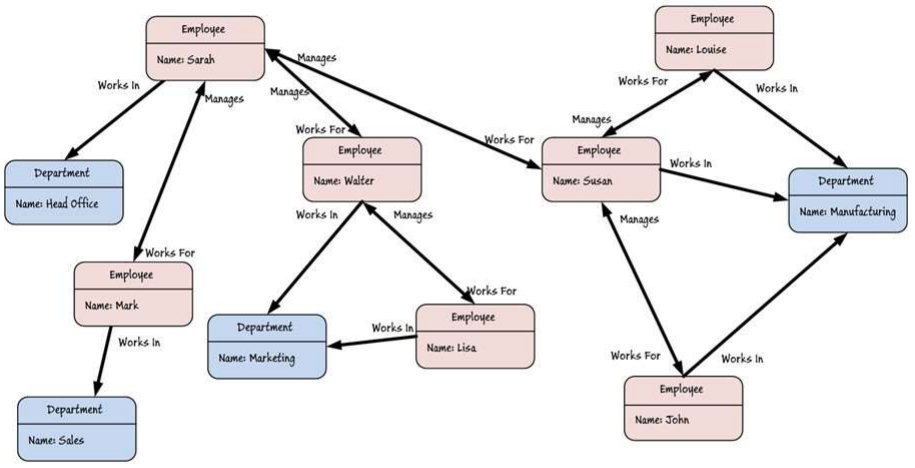
GRAPH DATABASE EXAMPLE



GRAPH DATABASE EXAMPLE

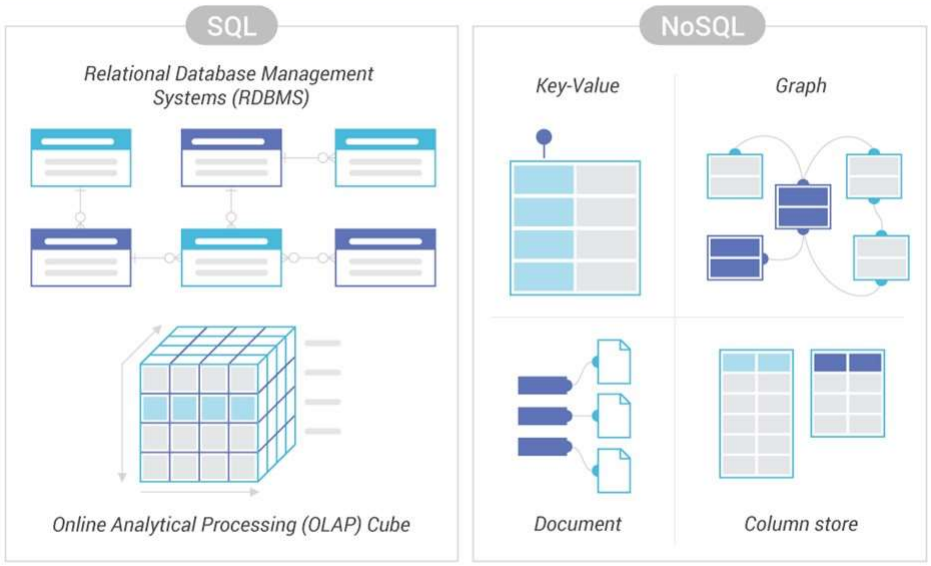


GRAPH DATABASE EXAMPLE

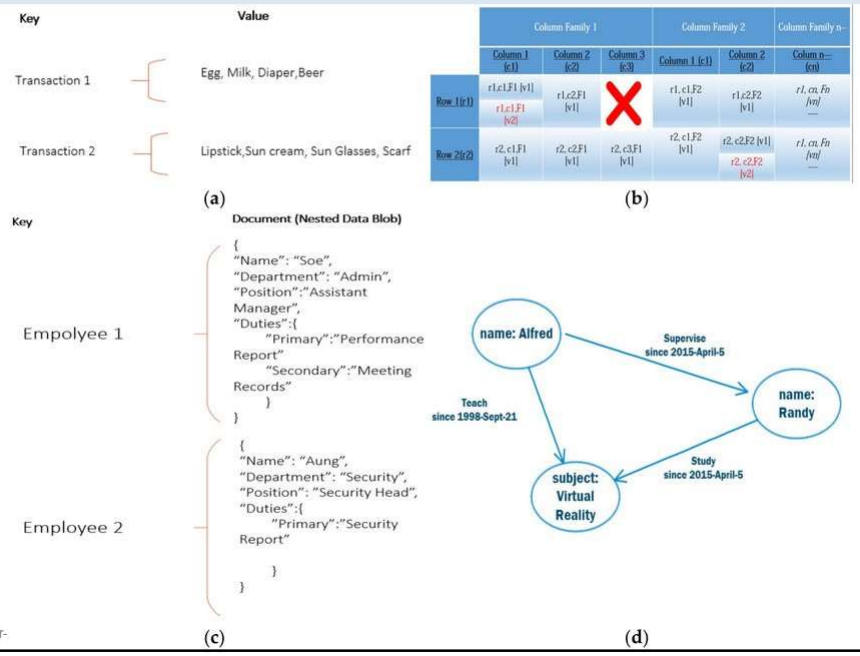


39

PERSISTENCE LAYER DESIGN-SUMMARY



PERSISTENCE LAYER DESIGN-SUMMARY



PERSISTENCE LAYER DESIGN-SUMMARY

1. **Relational database** — Represents the database as a collection of relations. A relation is nothing but a table (rows & columns) of values.
2. **Key-value store** — Stores data with simple indexed keys and values. Examples include Oracle NoSQL database, Redis, Aerospike, Oracle Berkeley DB, Voldemort, Amazon DynamoDB and Infinity DB.
3. **Document database** — A more complex and structured version of the key-value model, which gives each document its own retrieval key. Examples include Orient DB, MarkLogic, MongoDB, IBM Cloudant, Couchbase, and Apache CouchDB.
4. **Wide column store** — Uses tables, rows and columns. But the format and naming of the columns can vary in different rows within the same table. Examples include Apache Cassandra, Scylla, Datastax Enterprise, Apache HBase, Apache Kudu, Apache Parquet and MonetDB.
5. **Graph database** — Presents interconnected data as a logical graph. Examples include Neo4j, JanusGraph, FlockDB and GraphDB.

VARIOUS NAMES FOR SAME CONCEPTS

Theory	Database	File	SOM	ER
Relation	Table	File	Class	Entity Set
Tuple	Row	Record	Object	Entity
Attribute	Column	Field	Attribute	Attribute


```
If(anyQuestions)
{
 askNow();
}
else
{
 thankYou();
 submitAttendance();
 endClass();
}
```

REFERENCES

1. ***Software Architecture***, *Perspectives on an Emerging Discipline By Mary Shaw & David Garlan*
2. ***The Art of Software Architecture***, *Design Methods & Techniques By Stephen T. Albin*
3. ***Essential Software Architecture***, *By Ian Gorton*
4. ***Microsoft Application Architecture Guide***, *By Microsoft*
5. ***Design Patterns***, *Elements of Reusable Object-Oriented Software By by Erich Gamma, Richard Helm, Ralph Johnson & John Vlissides*
6. ***Refactoring, Improving the Design of Existing Code***, *By Martin Fowler & Kent Beck*