

Introduction to Management Information Systems

Databases

Data Resource Management

Learning objectives

- ▶ understand what is a database and why is used to store data;
- ▶ understand how a relational database uses keys and relationships to link data between tables
- ▶ explain entity relationship (ER) diagrams and how they are used for database design
- ▶ explain the use of SQL for both data definition and data manipulation

What is a database?

A database is a logically organized collection of related data designed and built for a specific purpose

databases

- ▶ large companies stored large quantities of data
- ▶ very expensive computerized systems
- ▶ storage systems

then

- ▶ home users (PCs)
- ▶ store data in files, in folders
- ▶ store data in spreadsheets

databases

- ▶ small business (SME)
 - ▶ employees
 - ▶ stock
 - ▶ sales
 - ▶ customers
- ▶ or just for personal use e.g. hobbies

databases

need

- ▶ small scale database management system (e.g. Access)
- ▶ cost effective
- ▶ easy to use
- ▶ suitable for a purpose
- ▶ scalable
- ▶ web-enabled
- ▶ use with programming languages

relational databases

- ▶ an organized collection of data in tables
- ▶ a table has columns and rows
- ▶ a row - a record
 - ▶ e.g. a book, a student
- ▶ a column - a detail about a record
 - ▶ e.g. ISBN, title, author, student number, student name, student date of birth

relational databases

create and maintain tables

- ▶ use a **Data Definition Language (DDL)**
 - ▶ e.g. Structured Query Language (SQL)

access data

- ▶ use a **Data Manipulation Language (DML)**
 - ▶ e.g. Structured Query Language (SQL)

relational databases

- ▶ interact with the database
- ▶ perform database operations
- ▶ add, update, delete, retrieve
- ▶ use SQL
- ▶ use an SQL engine (inside an DBMS)
- ▶ e.g. SQLite, MySQL

web databases

- ▶ example - an online retailer
 - ▶ HTML documents
 - ▶ stock, prices, catalog, inventory control
 - ▶ ordering processing system
- ▶ web application
 - ▶ user query & orders
 - ▶ application takes query, connects with the ordering system
 - ▶ connects with the database

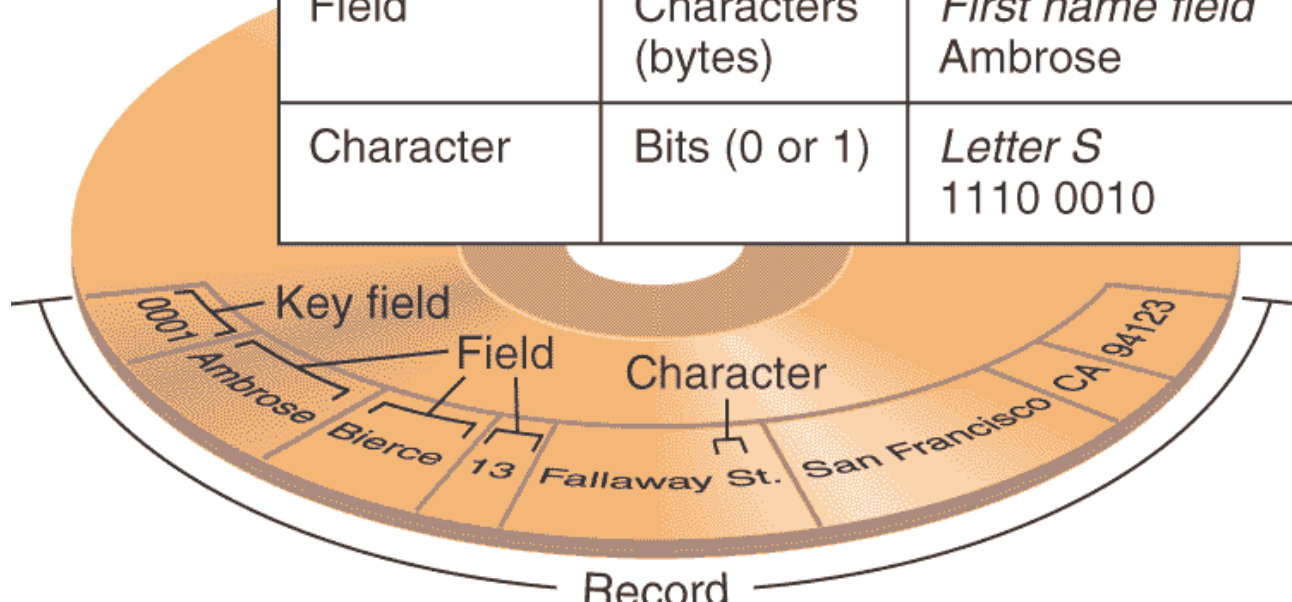
Database Systems

Data is stored hierarchically for easier storage and retrieval

- **Database** has tables or files
- **Files (Tables)** collections of related records
- **Records** collections of related fields
- **Field** unit of data containing 1 or more characters
- **Character** a letter, number or special character
- **Bit** made of bits
- **Bit** 0 or 1

data hierarchy

Type of data	Contains	Example
Database	Several files	<i>Your personal database</i> Friends' addresses file, CD titles file, Term papers file, etc.
File	Several records	<i>Friends' addresses file</i> Bierce, Ambrose 0001; London, Jack 0234; Stevenson, Robert L. 0081; etc.
Record	Several fields	<i>Ambrose Bierce's name and address</i> 13 Fallaway St. San Francisco, CA 94123
Field	Characters (bytes)	<i>First name field</i> Ambrose
Character	Bits (0 or 1)	<i>Letter S</i> 1110 0010

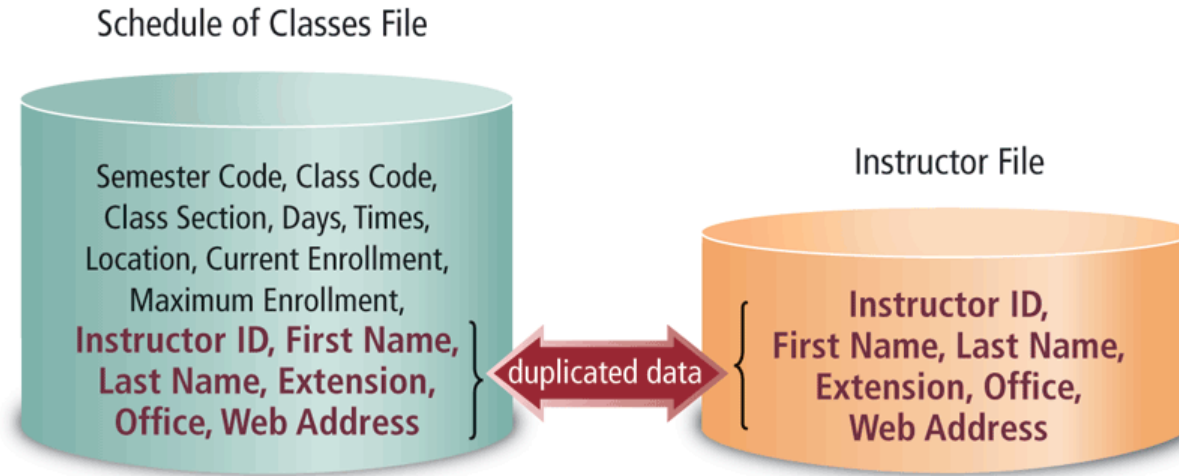


Primary Keys

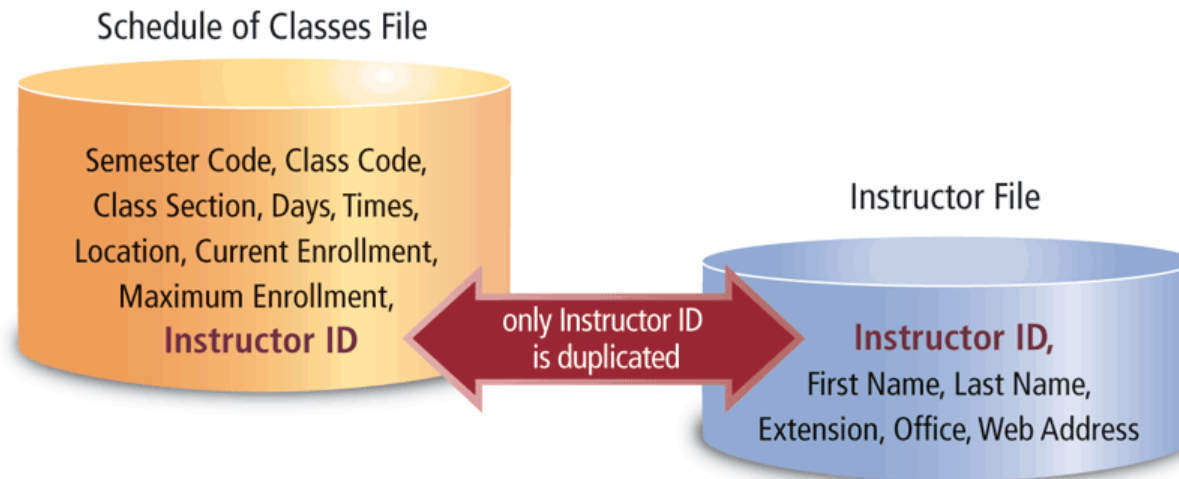
- **Primary keys** must be **unique**
- identifies a record
- e.g.
 - **passport number**
 - **student id number**
 - social security number
- Numbers
 - faster access

File Processing Versus Databases

File Processing Example



Database Example



Foreign Keys

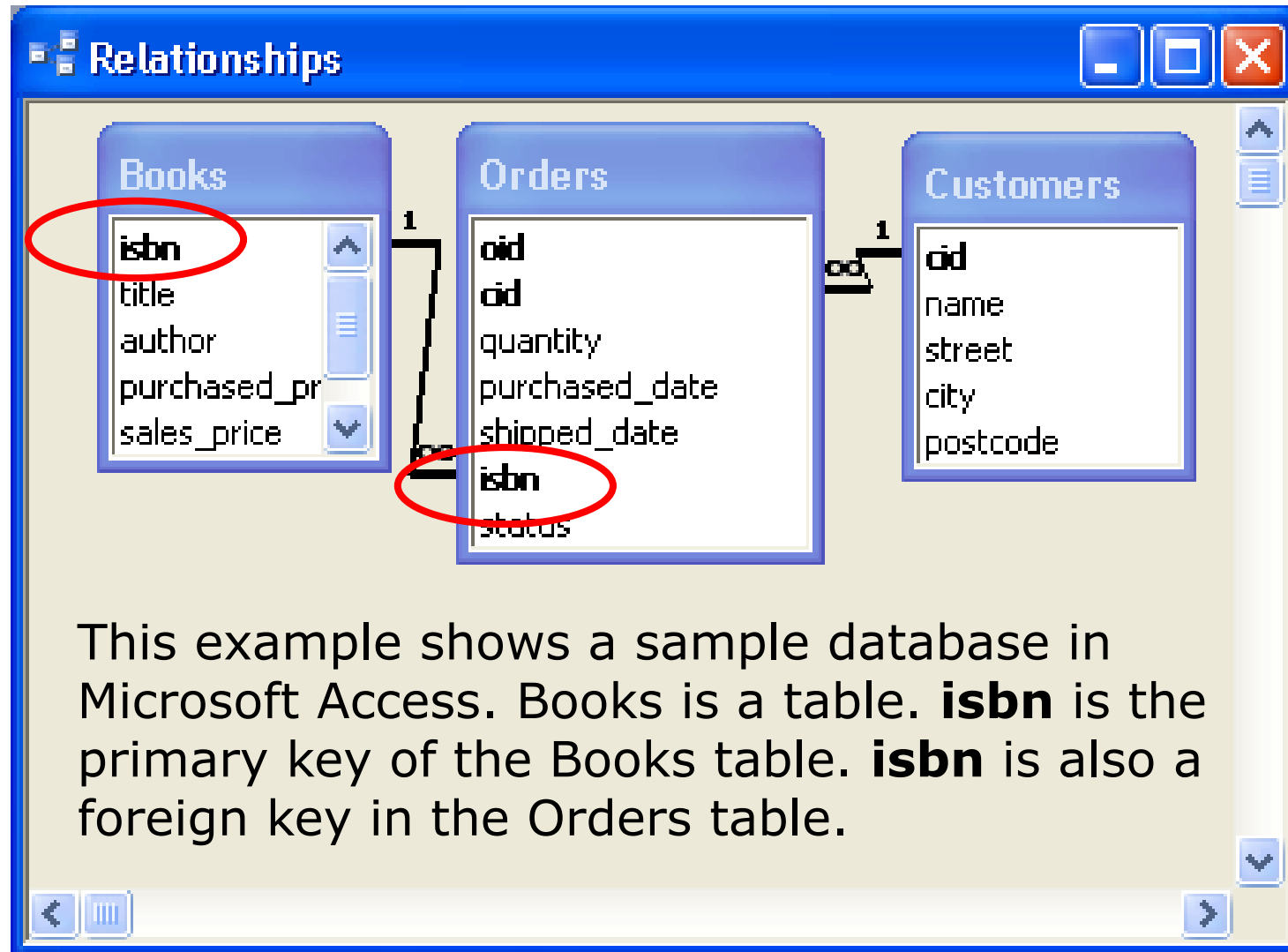
Foreign keys are primary keys in another table

Keys represent the relationship between tables

e.g.

- course table
- foreign key = student id number
- student id number is the primary key in the student table
- Same format in both tables (referential integrity)

Managing Files: Basic Concepts



Database Management System (DBMS)



The diagram consists of four green cylinders arranged in a 2x2 grid. Each cylinder contains a function of a Database Management System (DBMS). The cylinders are connected by thin white lines, forming a square shape. The background is white with purple geometric shapes on the right side.

Create a computerized database

Add, modify, and delete data

Sort and retrieve data

Create forms and reports from the data

Database Management Systems

Reduced data redundancy

- No duplication

Improved data integrity

- Means the data is accurate, up to date

Increased security

- Limit who can create, read, update, and delete

Ease of data maintenance

- validation, backup,
- procedures for data inserting, updating, and deletion

Database Management Systems

Popular Database Management Systems

Database	Manufacturer	Computer Type
Access	Microsoft Corporation	Personal computer, server, mobile devices
Adabas	Software AG	Server, mainframe
D ³	Raining Data	Personal computer, server
DB2	IBM Corporation	Personal computer, server, mainframe
Essbase	Oracle Corporation	Personal computer, server, mobile devices
FastObjects	Versant Corporation	Personal computer, server
FileMaker	FileMaker, Inc.	Personal computer, server
GemFire	GemStone Systems	Server
Informix	IBM Corporation	Personal computer, server, mainframe
Ingres	Ingres Corporation	Personal computer, server, mainframe
InterBaseSMP	Embarcadero Technologies	Personal computer, server
KE Texpress	KE Software, Inc.	Personal computer, server
MySQL	Oracle Corporation	Personal computer, server
ObjectStore	Progress Software Corporation	Personal computer, server
Oracle Database	Oracle Corporation	Personal computer, server, mainframe, mobile devices
SQL Server	Microsoft Corporation	Server, personal computer
SQL Server Compact Edition	Microsoft Corporation	Mobile devices
Sybase	Sybase Inc.	Personal computer, server, mobile devices
Teradata Database	Teradata	Server
Versant	Versant Corporation	Personal computer, server
Visual FoxPro	Microsoft Corporation	Personal computer, server

SQLite

Database Administrator (DBA)

Ensures the database's

- Recoverability
- Integrity
- Security
- Availability
- Reliability
- Performance

Database Models

Hierarchical database

- ▶ Fields or records are arranged in a family tree, with child records subordinate to parent or higher-level records

Network database

- ▶ Like a hierarchical database, but each child record can have more than one parent record

Relational database

- ▶ Relates, or connects, data in different files using a key

Object-oriented database

- ▶ Uses objects (software written in small, reusable chunks) as elements within database files

Multidimensional database

- ▶ Models data as facts, dimensions, or numerical measures for use in the interactive analysis of large amounts of data

Data Models

Conceptual Data Model: what the system contains

Logical Data Model: how the system should be implemented regardless of the DBMS

Physical Data Model: how the system will be implemented using a specific DBMS system.

Database Example: Microsoft Access

Outline

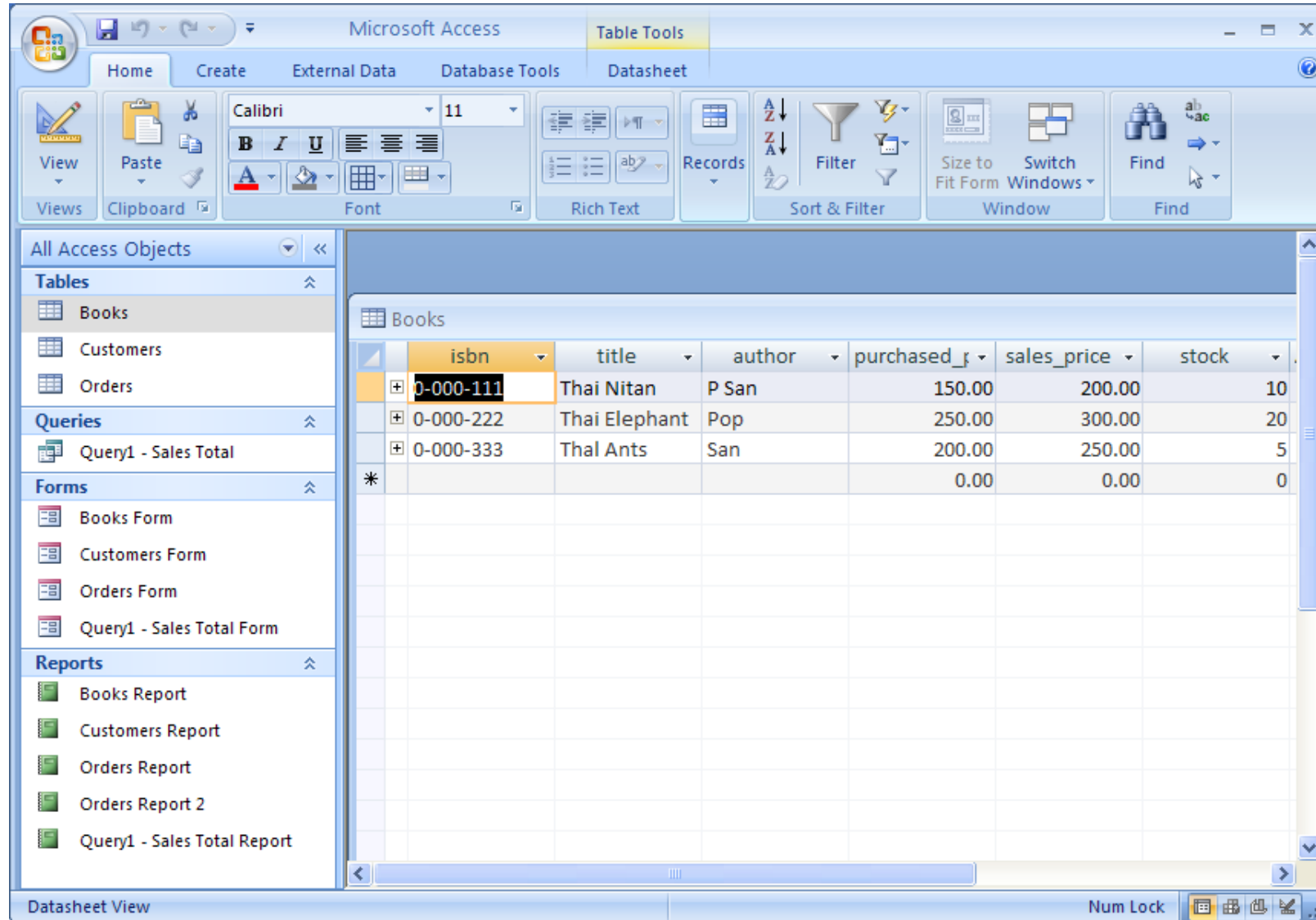
Microsoft Access Objects:

1. Tables
2. Queries

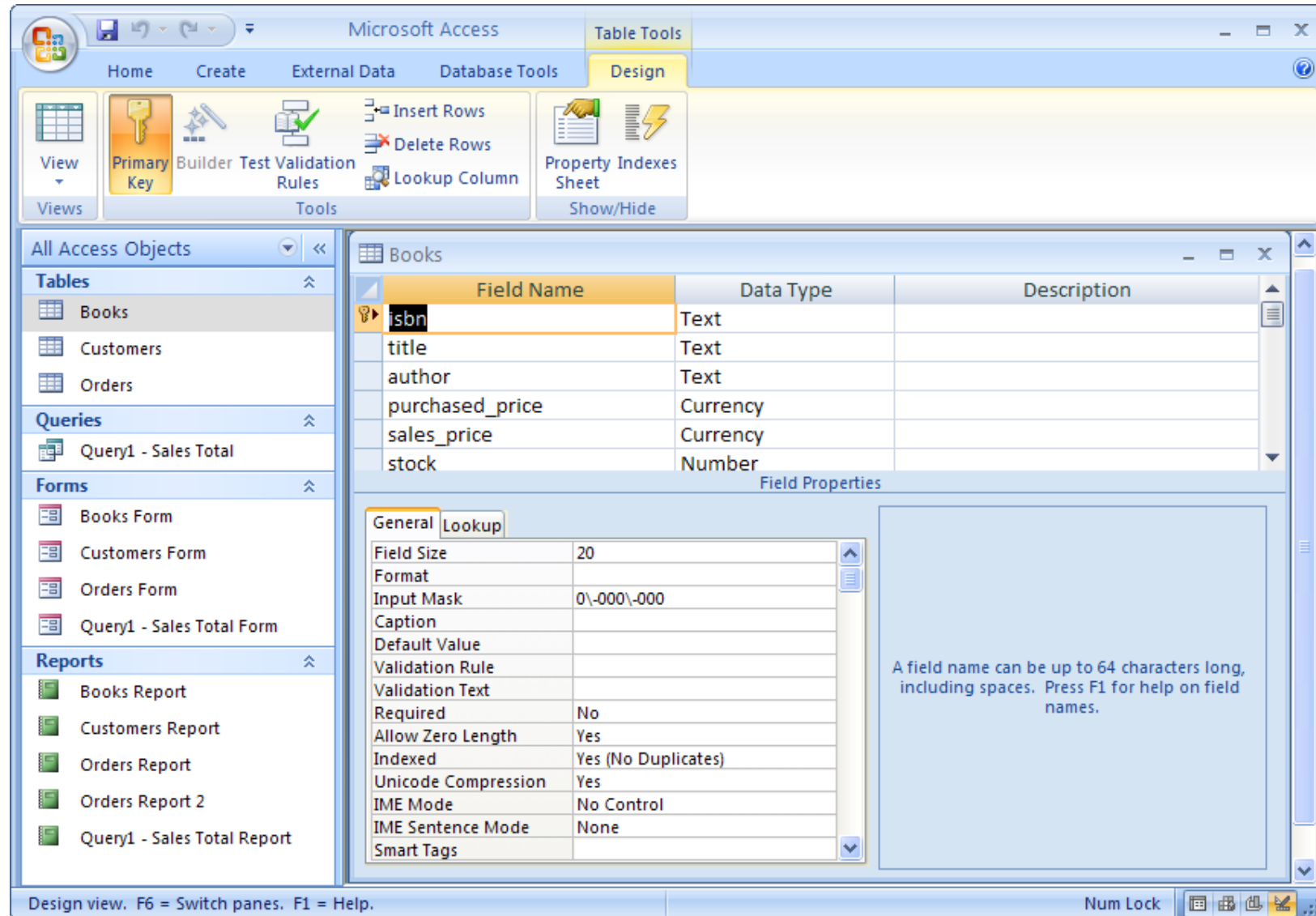
also has

- ▶ Forms
- ▶ Reports

The Database Window

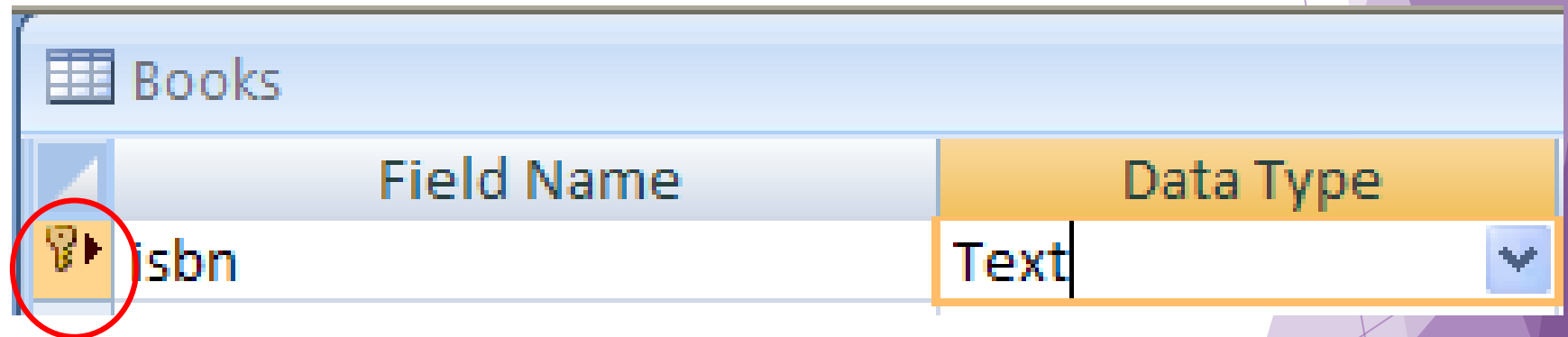


Design View



Primary Key

- ▶ Every record must have a **primary key**
- ▶ a different value for that particular field
- ▶ will be noted with a key image to the left



The screenshot shows a database table named 'Books'. The table has two columns: 'Field Name' and 'Data Type'. The first row shows the field 'isbn' with a data type of 'Text'. A yellow key icon is positioned to the left of the 'isbn' field name, and this icon is circled in red to indicate it is the primary key.

Field Name	Data Type
isbn	Text

Data

AutoNumber - automatically assigned **unique integer**

Field Name -

- should represent the contents of the field such as “isbn”, “title”, “author”
- not exceed 64 characters in length and may include spaces

Data Types

Text - default type

- allows any combination of letters and numbers up to a max of 255 characters

Number - any number

Date/Time - A date, time, or both

Yes/No - true/false, yes/no, on/off, or other boolean values

Memo - A text type

- stores up to 64,000 characters

Currency - monetary values

- can include a dollar sign (\$)
- can set decimal and comma positions

Number field size

Byte integers between 1 - 255

Integer integers between -32,768 and 32,768

Long Integer (default)

integers between -2 billion and 2 billion

Single or float - single-precision floating-point number

- ▶ up to 7 dp (32 bits)

Double - double-precision floating-point number

- ▶ up to 15 dp (64 bits)

Decimal - number that allows decimal places

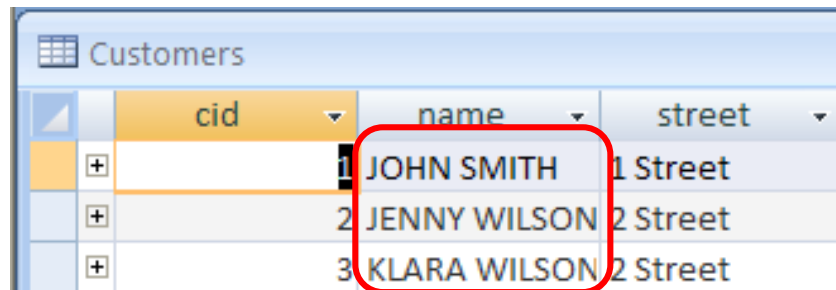
- ▶ up to 28 dp
- ▶ bigger numbers than long integer
- ▶ memory usage high
- ▶ errors e.g. order

Field Format

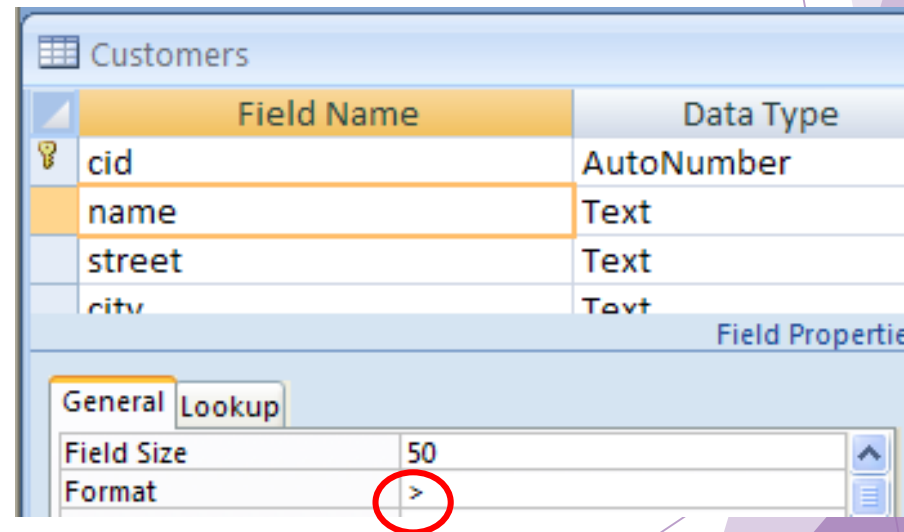
Symbol	Description
<	Force all characters to lowercase
>	Force all characters to uppercase

Field Format

- Force all characters in the ***Name*** field to be display in uppercase.



	cid	name	street
+		1 JOHN SMITH	1 Street
+		2 JENNY WILSON	2 Street
+		3 KLARA WILSON	2 Street



Field Name	Data Type
cid	AutoNumber
name	Text
street	Text
city	Text

Field Properties	
General	Lookup
Field Size	50
Format	>

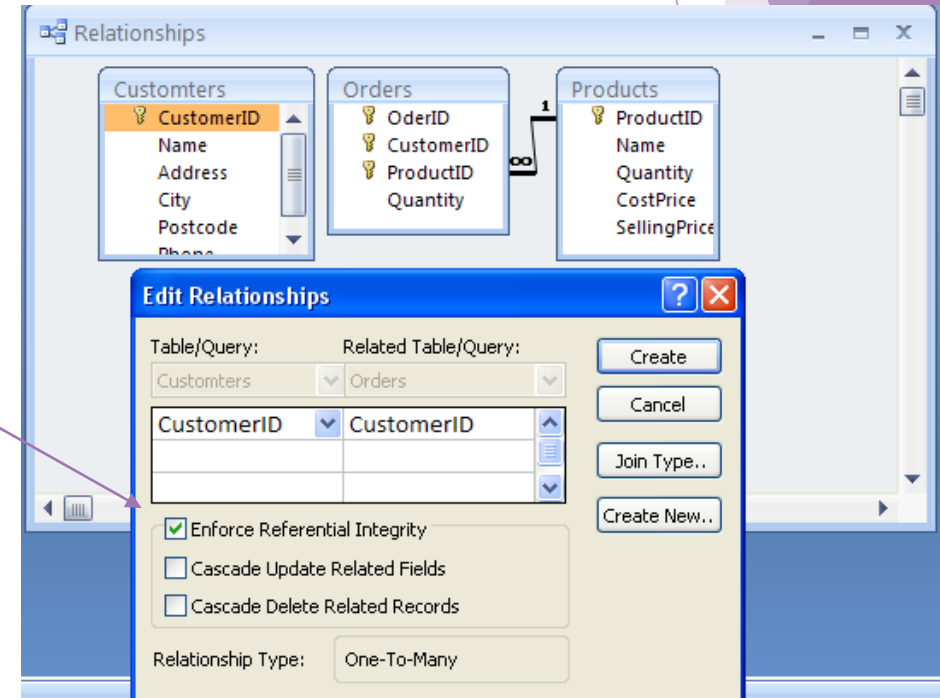
Input Mask

Character	Description	Input Mask	Sample Value
0	Digit (0 to 9, entry required, plus [+] and minus [–] signs not allowed)	(000) 000-0000	(206) 555-0248
9	Digit or space (entry not required, plus and minus signs not allowed)	(999) 999-9999	(206) 555-024

Relationships

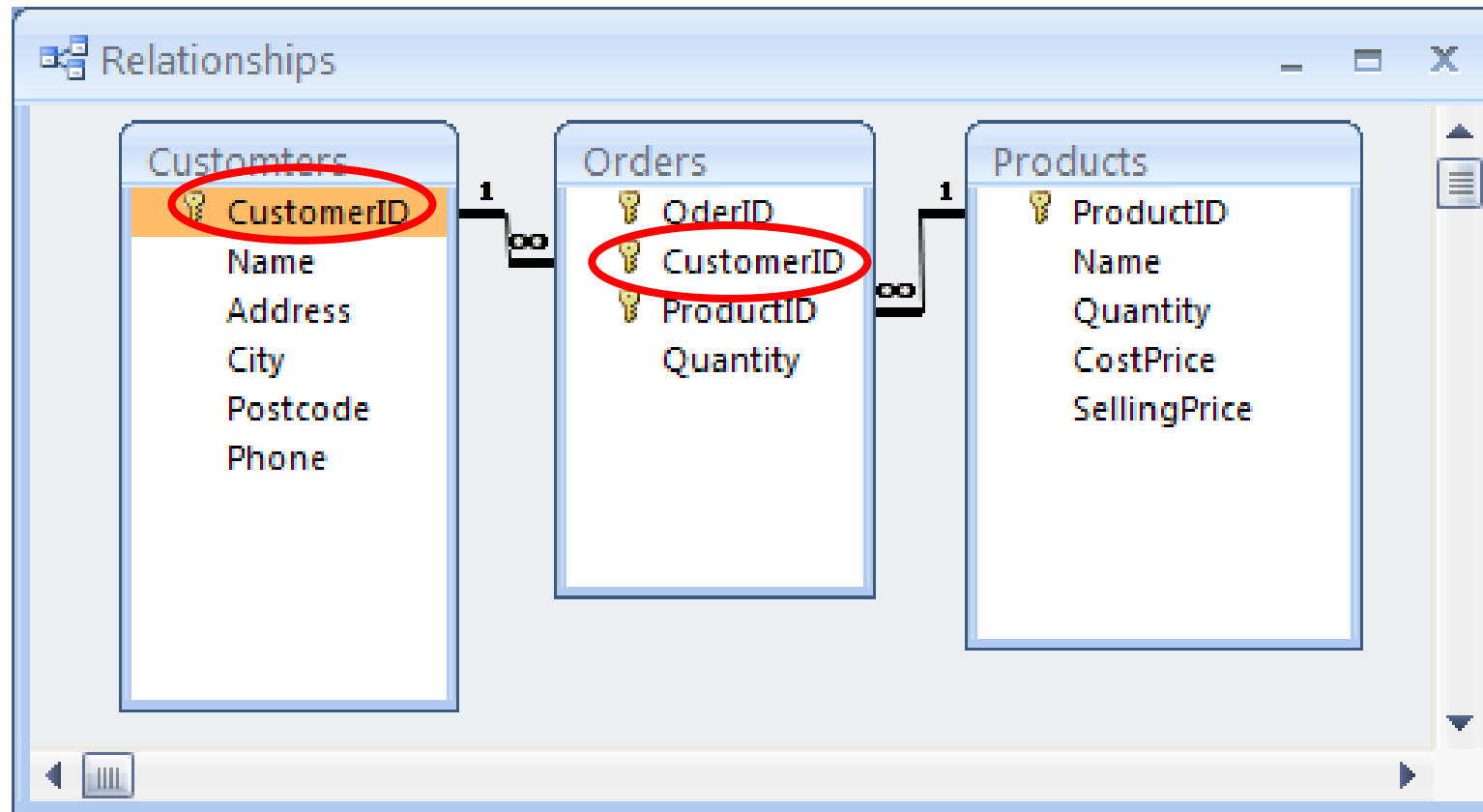
Enforce Referential Integrity

- ▶ ensures that the relationships are valid and
- ▶ data is not deleted when data is added, edited, or deleted



Relationships

7. A line now connects the two fields in the Relationships window



Database Management Systems

A query language

- ▶ consists of simple, English-like statements that allow users to specify the data to display, print, or store

Query by Example (QBE)

- ▶ provides a GUI to assist users with retrieving data

Query by example

Orders Query

The diagram shows three tables: Customers, Orders, and Products. Customers has fields: Custom (key), Name, Address, City, and Postal. Orders has fields: OderID (key), CustomerID (key), ProductID (key), and Quantity. Products has fields: Product (key), Name, Quantit, CostPric, and SellingP. Relationships are shown as 1-to-many: Customers (1) to Orders (many), and Orders (1) to Products (many).

Field:	OderID	CustomerID	ProductID	Quantity
Table:	Orders	Orders	Orders	Orders
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:				
or:				

Structured Query Language (SQL)

Data Definition Language

- ▶ A DDL is a language used to define data structures and modify data.
- ▶ used to add, remove, or modify tables within in a database
- ▶ used in database applications
- ▶ considered a subset of SQL
- ▶ can be other than SQL

tables

```
CREATE TABLE items_ordered (  
  orderID      integer      primary key,  
  customerID   float,  
  order_date   timestamp,  
  item         varchar ( 255 ),  
  quantity     float       check (quantity>=0),  
  price        float,  
  FOREIGN KEY (customerid) REFERENCES customers(customerid)  
);
```

constraints

```
CREATE TABLE myemployee (  
    ID integer primary key autoincrement,  
    firstname varchar ( 30 ),  
    lastname varchar ( 50 ),  
    title   varchar ( 30 ),  
    age     number ( 3 ) check (age>=18),  
    salary  number ( 10 , 2 ) check (salary >= 10000)  
);
```

Data Manipulation Language

- ▶ deals with data manipulation
- ▶ used to store, modify, retrieve, delete and update data in a database
- ▶ includes most common SQL statements
- ▶ e.g. SELECT, INSERT, UPDATE, DELETE

Select

SELECT	<attribute list>
FROM	<table list>
WHERE	<condition>

<i>what you want to see</i>	<i>(columns)</i>
<i>from where</i>	<i>(tables)</i>
<i>which rows</i>	<i>(rows e.g. where id>100)</i>

INSERT

INSERT into employee

values ('richard','k','marini', '653298653', '30-dec-52','98 oak forest,katy,tx', 'm', 37000,'987654321', 4)

INSERT into employee (fname, lname, ssn)
values ('richard', 'marini', '653298653')

UPDATE

```
UPDATE Employees  
SET City = 'Newark'  
WHERE ID = 4;
```

```
UPDATE Employees  
SET Salary = Salary * 1.1  
WHERE Age > 25;
```

DELETE

DELETE from employee where id=101;

DELETE from employee;

not same as
REMOVE employee

Referential integrity

Items_ordered uses a customer id that **does not exist**

SQL error: foreign key constraint failed

Update table with an **incorrect value from another table**

SQL error: foreign key constraint failed

Items_ordered has items by a customer, and you **try to delete the customer**

SQL error: foreign key constraint failed

Delete the items_ordered by a customer **then** the customer

noSQL

noSQL

Not only SQL

Most NOSQL systems are distributed databases or distributed storage systems

- Focus on semi-structured data storage, high performance, availability, data replication, and scalability

Introduction

- ▶ NOSQL systems focus on storage of “big data”
- ▶ Typical applications that use NOSQL
 - ▶ Social media
 - ▶ Web links
 - ▶ User profiles
 - ▶ Marketing and sales
 - ▶ Posts and tweets
 - ▶ Road maps and spatial data
 - ▶ Email

Introduction to NOSQL Systems (cont'd.)

NOSQL characteristics related to distributed databases and distributed systems

- ▶ Scalability
- ▶ Availability, replication, and eventual consistency
- ▶ Replication models
 - ▶ Master-slave
 - ▶ Master-master
- ▶ Sharding of files
- ▶ High performance data access

Introduction to NOSQL Systems (cont'd.)

- ▶ NOSQL characteristics related to data models and query languages
 - ▶ Schema not required
 - ▶ Less powerful query languages
 - ▶ Versioning

Categories of NOSQL Systems

- ▶ Document-based NOSQL systems
- ▶ NOSQL key-value stores
- ▶ Column-based or wide column NOSQL systems
- ▶ Graph-based NOSQL systems
- ▶ Hybrid NOSQL systems
- ▶ Object databases
- ▶ XML databases

NOSQL Systems

- ▶ Document stores
 - ▶ Collections of similar documents
- ▶ Individual documents resemble complex objects or XML documents
 - ▶ Documents are self-describing
 - ▶ Can have different data elements
- ▶ Documents can be specified in various formats
 - ▶ XML
 - ▶ JSON

XML & JSON

- Databases function as data sources for Web applications

HTML

- Used in static Web pages

XML, JSON

- Self-describing documents
- Dynamic Web pages

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Race date="2010-12-31" name="New Years Meet">
- <Course>
  <CourseName>The new track</CourseName>
  <Address>Track Road 123</Address>
</Course>
- <Horses>
- <Horse Name="Bonfire">
  <Value>5000</Value>
  <DateOfBirth>1988-01-02</DateOfBirth>
  <Gender>M</Gender>
</Horse>
- <Horse Name="Faithfull Dobbin">
  <Value>3500</Value>
  <DateOfBirth>1986-05-31</DateOfBirth>
  <Gender>F</Gender>
</Horse>
- <Horse Name="Pegasus">
  <Value>3000</Value>
  <DateOfBirth>1992-06-23</DateOfBirth>
  <Gender>M</Gender>
</Horse>
</Horses>
</Race>
```

JSON

- ▶ JavaScript Object Notation
- ▶ a lightweight data-interchange format
- ▶ easy for humans to read and write
- ▶ easy for machines to parse and generate

```
{
  "scores": [
    {
      "Away_Score": 2,
      "Away_Team": "Newcastle",
      "Home_Score": 2,
      "Home_Team": "Arsenal"
    },
    {
      "Away_Score": 2,
      "Away_Team": "Napoli",
      "Home_Score": 4,
      "Home_Team": "Liverpool"
    }
  ]
}
```

24.4 NOSQL Key-Value Stores

No query language

Key-value stores

- ▶ focus on
 - ▶ high performance,
 - ▶ availability, and
 - ▶ scalability

Can store

- ▶ structured, unstructured, or semi-structured data

24.4 NOSQL Key-Value Stores

- ▶ **Key:**
 - ▶ unique identifier
 - ▶ associated with a data item
 - ▶ Used for fast retrieval
- ▶ **Value:**
 - ▶ the data item itself
 - ▶ Can be string or array of bytes
 - ▶ Application interprets the structure

Relational Databases

great for

- ▶ managing data
- ▶ data consistency

but at a cost

- ▶ resource intensive
- ▶ don't scale well

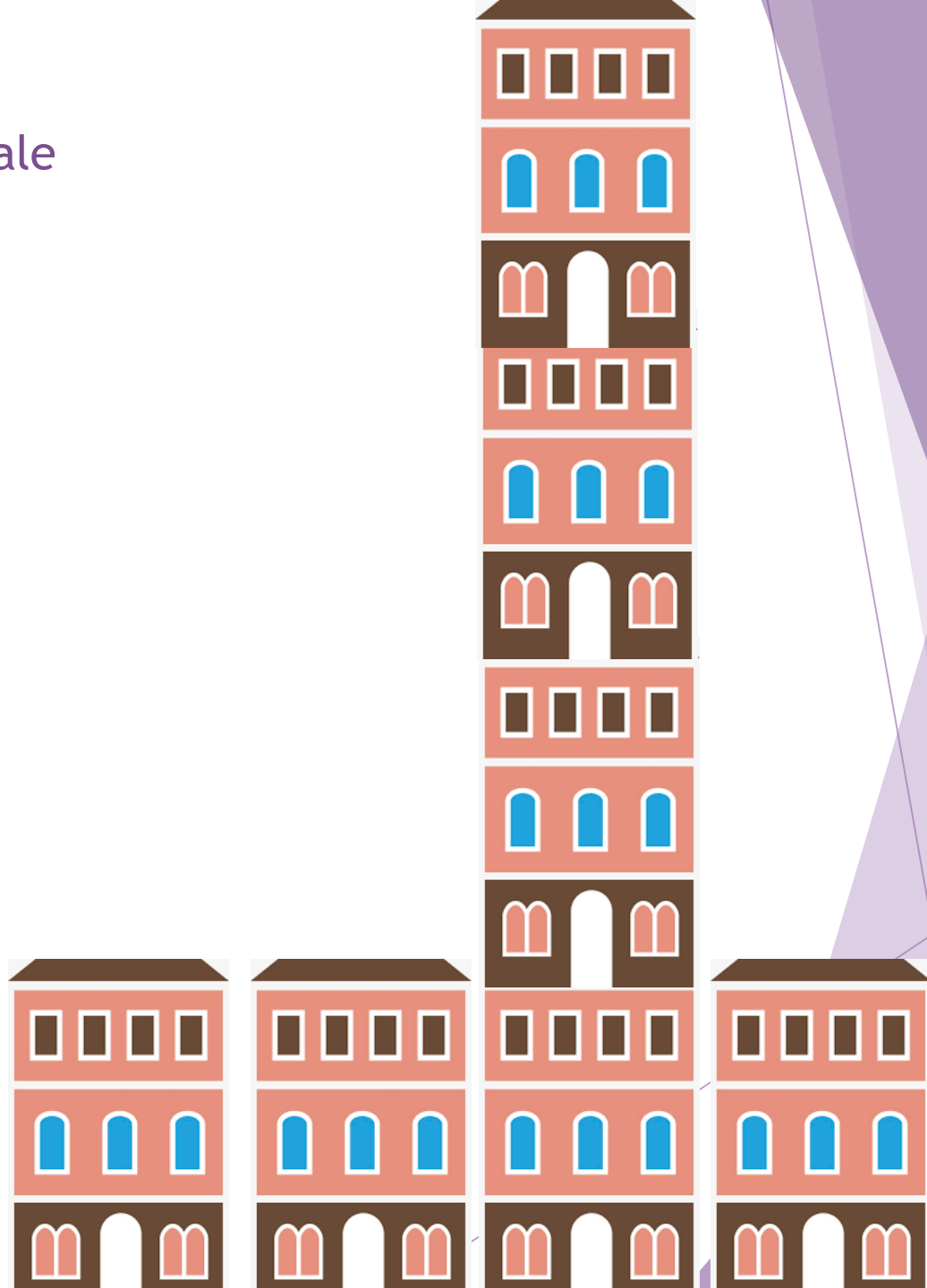
scale

- ▶ relational database
 - ▶ can scale horizontally



scale

- ▶ relational database
 - ▶ can scale horizontally
- ▶ noSQL
 - ▶ can scale horizontally
 - ▶ and vertically



why does nosql scale?

- ▶ data is independent
- ▶ remove relationships
- ▶ only uses 2 fields
 - ▶ key - value

Key	Value
194252165973	MacBook Pro 13"
42406659611	USB Microphone
36000341362	Hand Sanitizer
36196308002	Toilet paper

source: <https://www.youtube.com/watch?v=0buKQHokLK8>

why does nosql scale?

- ▶ data is independent
- ▶ remove relationships
- ▶ only uses 2 fields
 - ▶ key - value
 - ▶ can use JSON

Key	Value
194252165973	{ name: "MacBook Pro 13", price: \$1032.21, description: "laptop" }
42406659611	USB Microphone
36000341362	Hand Sanitizer
36196308002	Toilet paper

source: <https://www.youtube.com/watch?v=0buKQHokLK8>

partitions

- ▶ if large quantities of data
- ▶ split nosql database over servers
- ▶ these are now called **partitions**
- ▶ locate items using a hash value
 - ▶ 2 servers - split 100 into 2 - 1-50, 51-100
 - ▶ 4 servers - split 100 into 4 - 1-25, 26-50, 51-75, 76-100
 - ▶ 1-100 called a **keyspace**, can be 0-10000000+
- ▶ primary key transformed using a **hash function**

nosql Databases

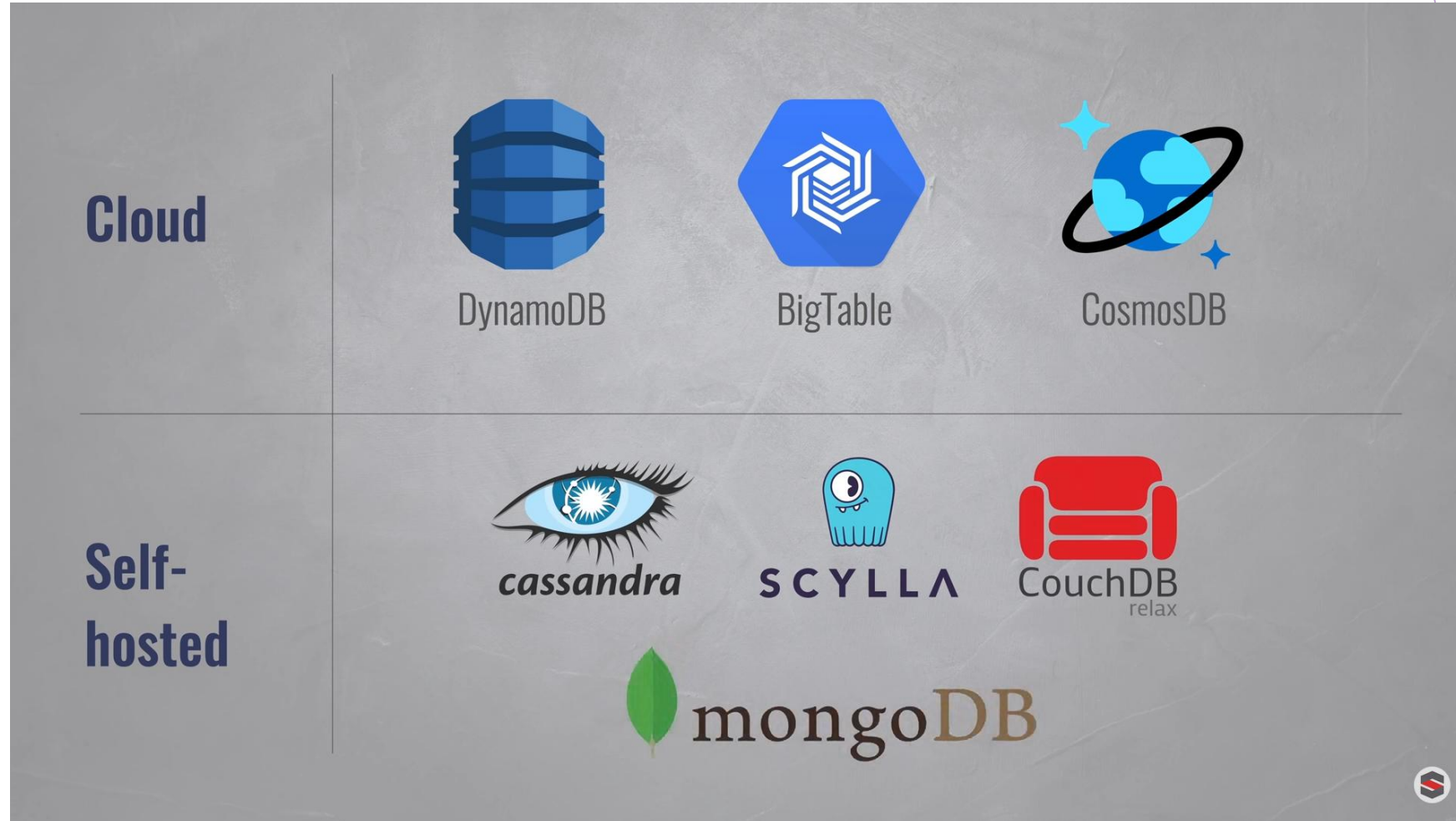
advantages

- ▶ data is structured in a relational databases
- ▶ nosql data is not limited to a structure
- ▶ data / database can evolve

disadvantage

- ▶ access data only by key
- ▶ no query option (efficiently)

examples



noSQL Systems

BigTable (Google)

- ▶ Column-based or wide column store

DynamoDB (Amazon)

- ▶ Key-value data store

Cassandra (Facebook)

- ▶ Uses concepts from both key-value store and column-based systems

noSQL Systems

MongoDB and CouchDB

- ▶ Document stores

Neo4J and GraphBase

- ▶ Graph-based NOSQL systems

OrientDB

- ▶ Combines several concepts

Database systems classified on the object model

- ▶ Or native XML model

The background features abstract, overlapping geometric shapes in various shades of purple, ranging from light lavender to deep, dark purple. These shapes are primarily located on the right side of the frame, creating a modern, layered effect.

Thank you!
any questions?