BOSTON UNIVERSITY

Research Computing

Introduction to Databases

Presented by

Yun Shen (yshen16@bu.edu)

Introduction

- What is Database
- Key Concepts
- Typical Applications and Demo
- Lastest Trends

What is Database

- Three levels to view:
  - Level 1: literal meaning – the place where data is stored
    - Database = Data + Base, the actual storage of all the information that are interested
  - Level 2: Database Management System (DBMS)
    - The software tool package that helps gatekeeper and manage data storage, access and maintenances. It can be either in personal usage scope (MS Access, SQLite) or enterprise level scope (Oracle, MySQL, MS SQL, etc).
  - Level 3: Database Application
    - All the possible applications built upon the data stored in databases (web site, BI application, ERP etc).

Examples at each level

- Level 1: data collection
  - text files in certain format: such as many bioinformatic databases
  - the actual data files of databases that stored through certain DBMS, i.e. MySQL, SQL server, Oracle, Postgresql, etc.
- Level 2: Database Management (DBMS)
  - SQL Server, Oracle, MySQL, SQLite, MS Access, etc.
- Level 3: Database Application
  - Web/Mobile/Desktop standalone application - e-commerce, online banking, online registration, etc.

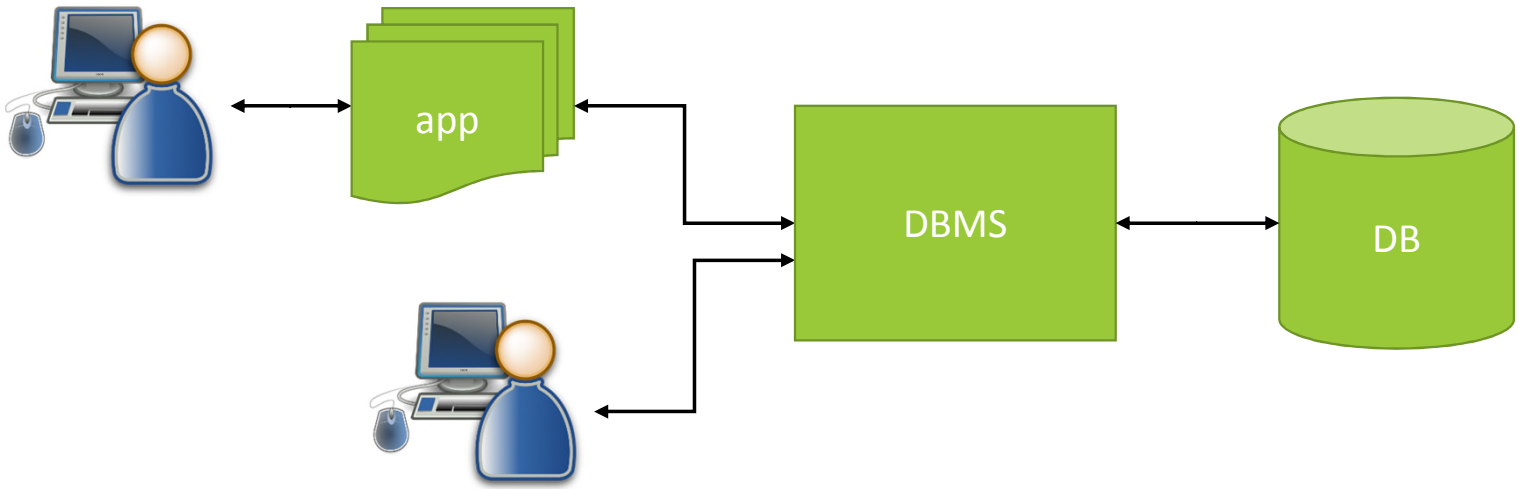
Examples at each level

- Level 1: data collection
  - ~~text files in certain format: such as many bioinformatic databases~~
  - the actual data files of databases that stored through certain DBMS, i.e. MySQL, SQL server, Oracle, Postgresql, etc.
- Level 2: Database Management System (DBMS)
  - SQL Server, Oracle, MySQL, SQLite, MS Access, etc.
- Level 3: Database Application
  - Web/Mobile/Desktop standalone application - e-commerce, online banking, online registration, Wikipedia, etc.

Database Types

- Flat Model
- Navigational databases
  - Hierarchical (tree) database model
  - Network/Graph model
- Relational Model
- Object model
- Document model
- Entity–attribute–value model
- Star schema

Typical Database Application Architecture



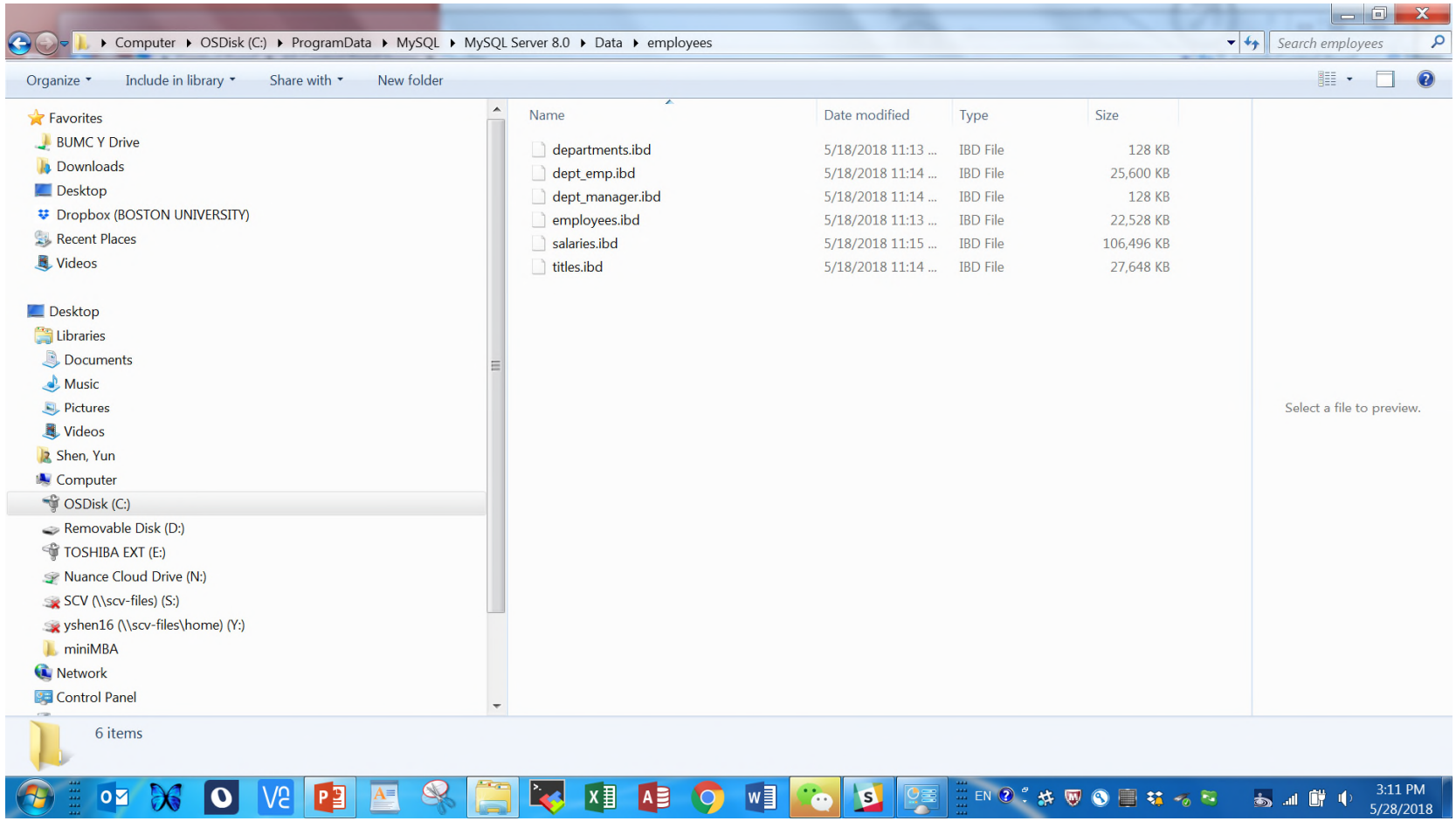
Data File Structure

- Demo :
  - Take a look at the following file directories:
    - MySQL -- C:\ProgramData\MySQL\MySQL Server 8.0\Data\
    - Access -- C:\ARCS_dbtutorial\db\access\
    - Postgresql -- /project/scv/examples/db/tutorial/data/postgresql/testdb/

BOSTON UNIVERSITY

Research Computing

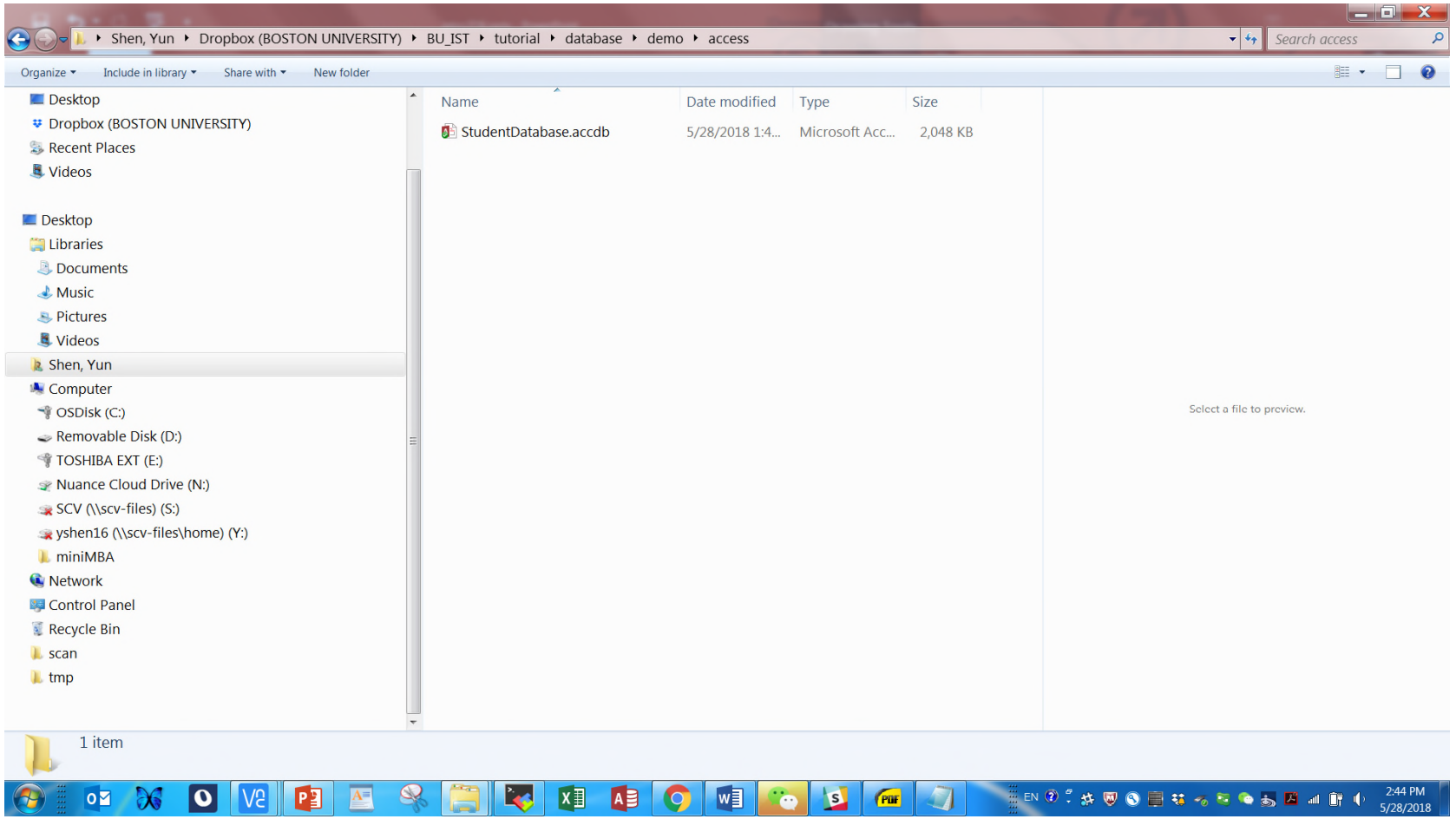
Data File Structure - MySQL



BOSTON UNIVERSITY

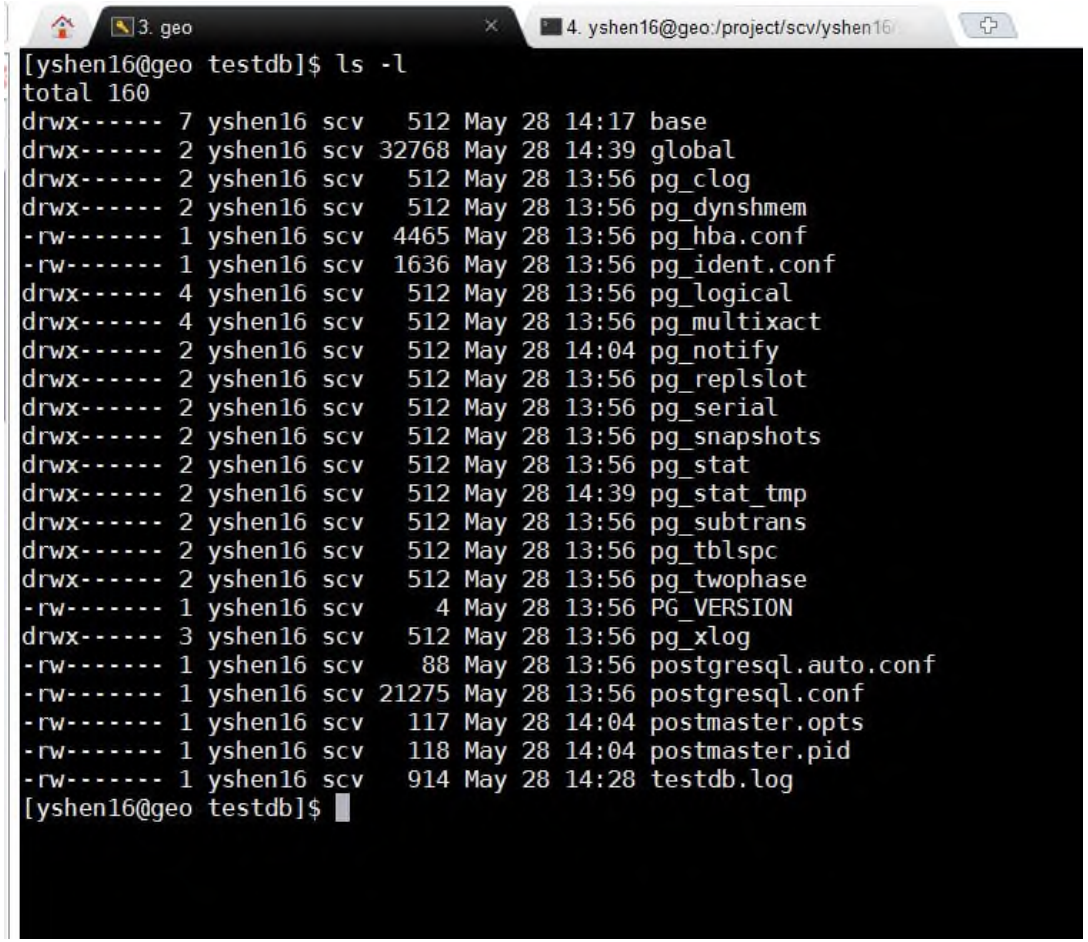
Research Computing

Data File Structure - Access



BOSTON UNIVERSITY Research Computing

Data File Structure - PostgreSQL



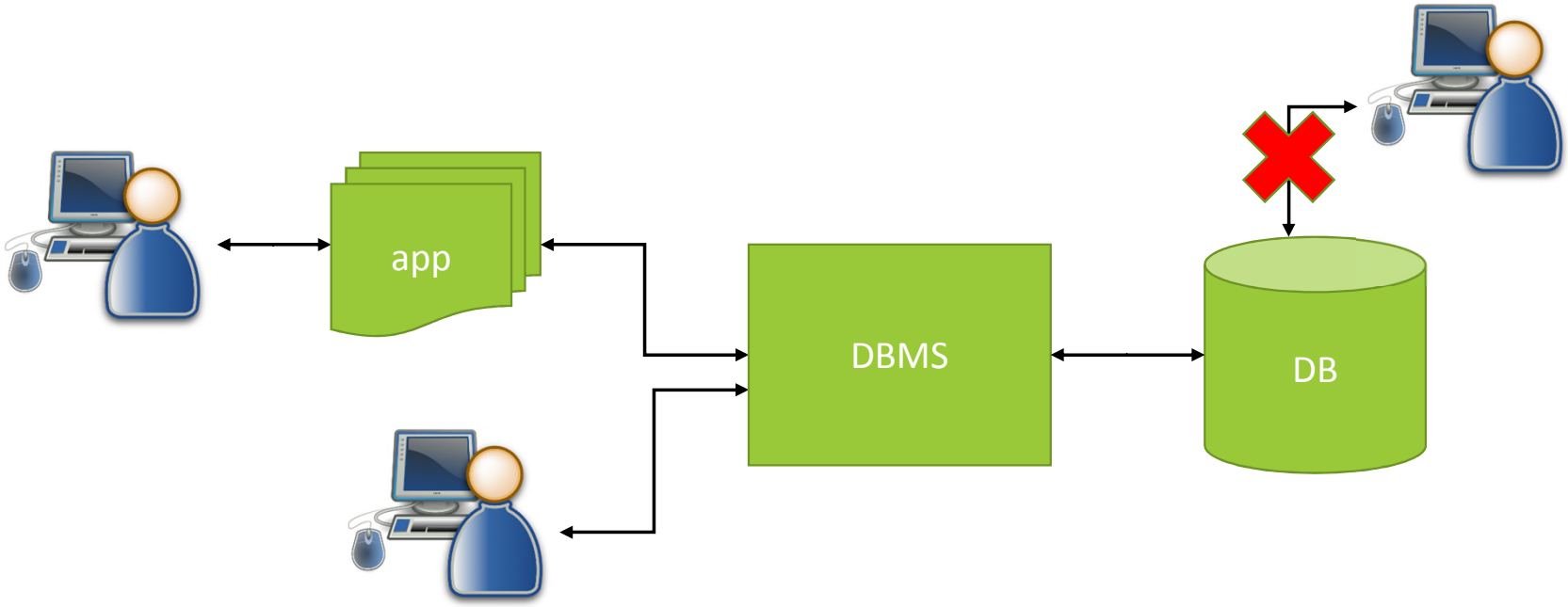
ATTENTION

!!!

NO database files can be accessed directly,

but only through the database engine, called “DBMS”

Typical Database Application Architecture



Three Common Acronyms

- SQL – Structured Query Language

- CRUD – Create, Read, Update, Delete
- ACID – Atomicity, Concurrency, Integrity and Durability (transaction)

Disadvantage of conventional flat file

- Redundancy - same data may store in many different copies
- Inconsistency – data regarding to same business entity may appear in different forms, for example, state name, phone number, etc. This made it hard to modify data and keep clean track of change; even loss of data
- Mixture of all data together – not clear in logical relationships between the data columns, thus hard to understand and manage once the data structure gets complex
- Hard to maintain and manage
- No concurrency support (only one can operate on the file)

First Acronym - ACID

Atomicity – transactions are either all or none (commit/rollback)
Consistency – only valid data is saved
Isolation – transactions would not affect each other
Durability – written data will not be lost

Good example : bank transaction

Most of challenges of ACID compliance come from multiple users/concurrent using of database

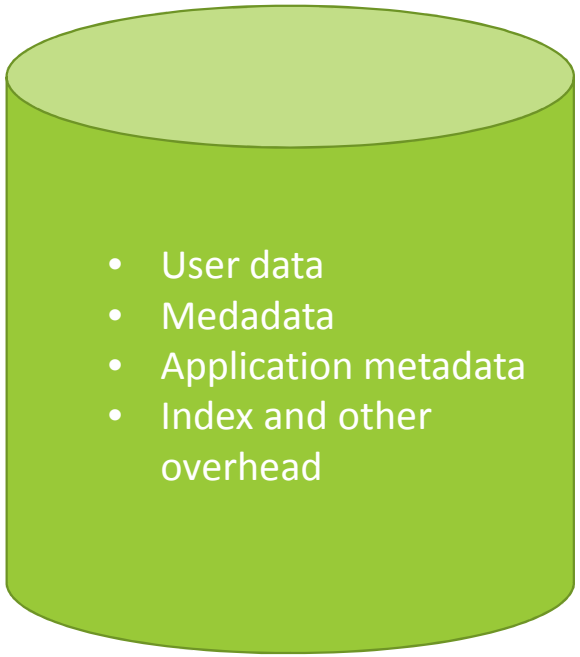
How Databases solves the problem?

- Self-describing data collection of related records (meta data, data about data) , detail explanation as below:

- Self-describing means:
  - Database not just contains data, but also contains definition of the structure of data, that can be considered ‘Meta data’; It includes many related info : table column definition, index and key info, constraints, etc
  - At database application level, database can also store other application related meta data as well, it makes personalization and customization of the application according to user profile much easier to handle. The typical example could be the user preference for those common social media sites or e-commerce sites, etc.

Database Content

Typical Database



- User Data: tables to store user data
- Meta data: keep the structure (schema) of the data, including table name, column name and type and contraints over the column(s)
- Application meta data: application specific meta data regarding to user settings or functions of the application
- Index and other overhead data: used for improving performance and maintenance, such as logs, track, security, etc.

Terminology and Concept – Tables (Relations)

The very central concepts of relational databases are Tables (Relations), Relationships.

Table (formally called ‘relation’) – is the building block of relational database. It stores data in 2D, with its row reflects one instance of record (tuple), and each of its column reflects one aspect of the attributes of all instances, column may also be called ‘field’.

For example, A ‘student’ table may contains (student id, first name, last name, grade, school name, home address, …), and each row may represent one student’s information, and each column of the table represents one piece of information of all students. And this is called a ‘relation’.

Primary Key and Foreign Key

- Primary key: Unique Identifier made of one or more columns to uniquely identify rows in a table. If the primary key contains more than one column, it can be called 'composite key' as well.
- Foreign Key: is the primary key of another table, which is referenced in the current table. It's the key to establish the relationship between the two tables, and through DBMS, to impose referential integrity.

Surrogate Key

- Surrogate key is a unique column
- added to a relation to use as the primary key when lack of natural column serves as primary key, or when composite key needs to be replaced for various reasons.
- Surrogate key is usually in form of auto increment numeric value, and of no meaning to the user, and thus cd often hidden in the table, or form or other entity for the internal use.
- Surrogate keys are often used in the place of composite key to add more flexibility to the table.

Terminology and Concept – E-R model

E-R Model: Entity-Relationship data model is the common technique used in database design. It captures the relationships between database tables and represent them in a graphical way. The relationships between two entities can be 1:1, 1:N, or M:N. And it is usually established through 'foreign key' constraint.

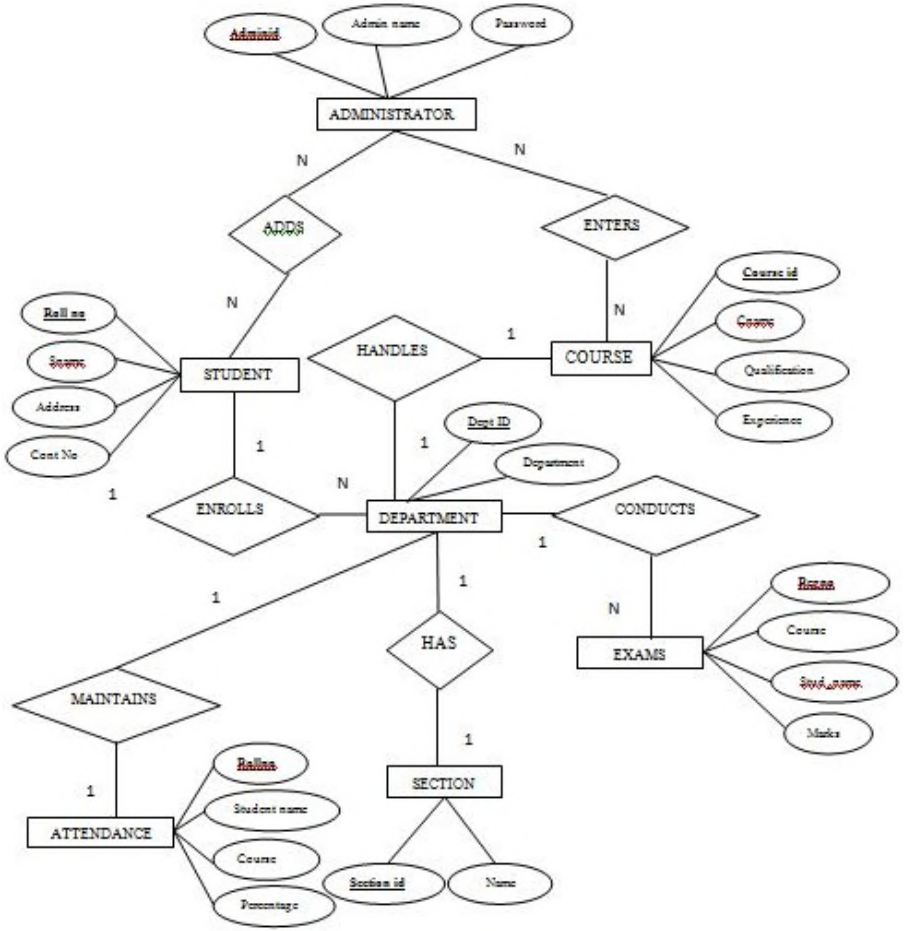
Examples:

1:1 Employee – Locker

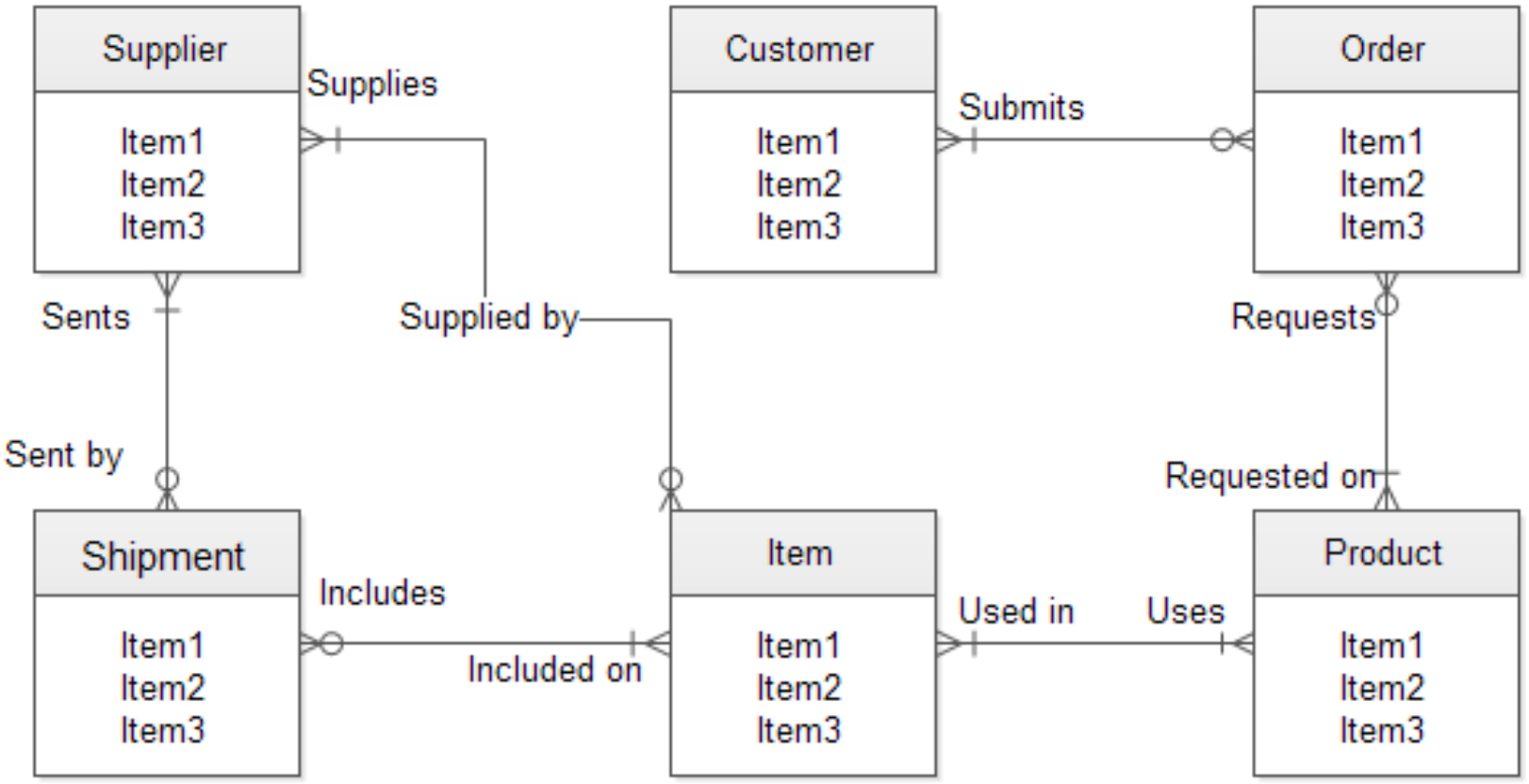
1:N Customer – Order, Order – Order Detail

M:N Student – Course

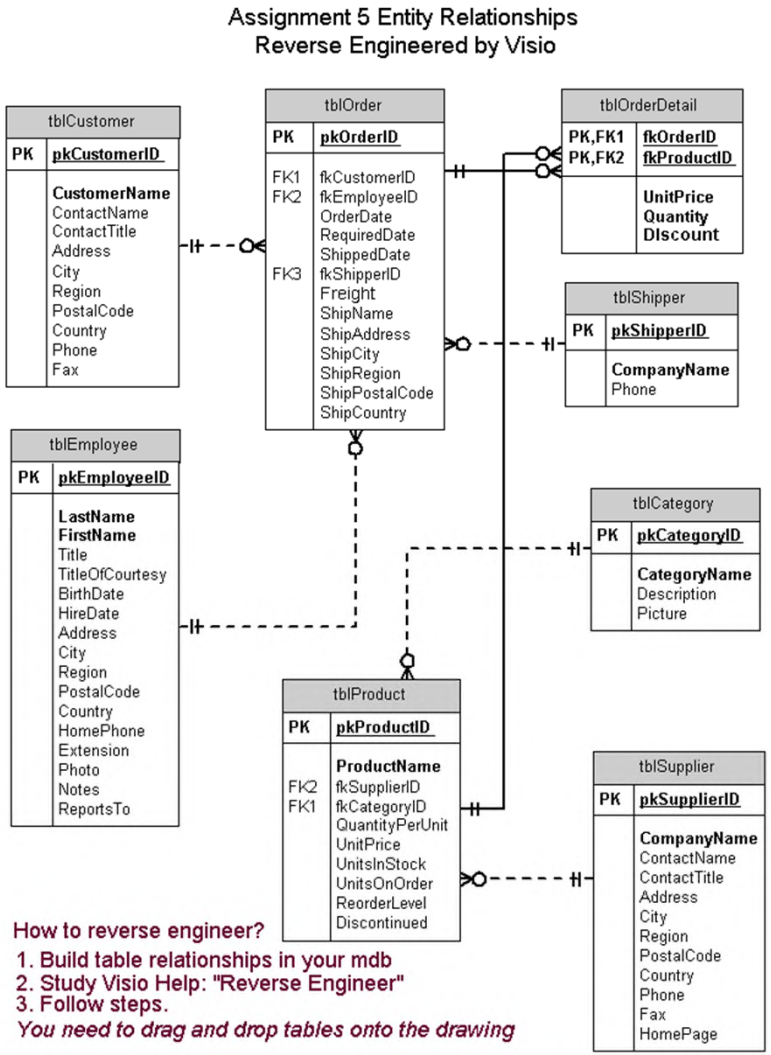
Sample E-R diagram #1



Sample E-R diagram #2



Sample E-R diagram #3



One more concept - Normalization

Wikipedia definition:

- **Database normalization**, or simply **normalization**, is the process of restructuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity. It was first proposed by Edgar F. Codd as an integral part of his relational model.
- Normalization entails organizing the columns (attributes) and tables (relations) of a database to ensure that their dependencies are properly enforced by database integrity constraints. It is accomplished by applying some formal rules either by a process of *synthesis* (creating a new database design) or *decomposition* (improving an existing database design).

Unnormalized Form (UNF) –

same attributes can be contained in one row

Example: Students take courses

Id	Name	Courses
1.	Jack	Mathematics Chemistry
2.	Tim	Chemistry
3.	Ana	Physics Chemistry

or

Id	Name	Course1	Course2
1.	Jack	Mathematics	Chemistry
2.	Tim	Chemistry	
3.	Ana	Physics	Chemistry

First Normal Form (1NF) –

each attributes can only have one single value in one row; no duplicated row; no row/column order

Example: Students take courses

Id	Name	Course
1.	Jack	Mathematics
4.	Jack	Chemistry
2.	Tim	Chemistry
3.	Ana	Physics
5.	Ana	Chemistry

Second Normal Form (2NF) –

1NF + no partial dependency (non-key attributes may not depend on any of candidate keys)

Example: Students take courses

Id	Name	Course	CourseID	Department
1.	Jack	Mathematics	M-1	Math
4.	Jack	Chemistry	C-1	Chemistry
2.	Tim	Chemistry	C-1	Chemistry
3.	Ana	Physics	P-1	Physics
5.	Ana	Chemistry	C-1	Chemistry

Second Normal Form (2NF) – continue

1NF + no partial dependency (non-key attributes may depend on any of candidate keys)

(Course) = Func(CourseID) (Department) = Func(CourseID)

Id	Name	CourseID
1.	Jack	M-1
4.	Jack	C-1
2.	Tim	C-1
3.	Ana	P-1
5.	Ana	C-1

Course ID	Course	Department
M-1	Mathematics	Math
C-1	Chemistry	Chemistry
P-1	Physics	Physics

Third Normal Form (3NF) –

2NF + no transitive dependency (non-key attributes may not depend on each other)

Example: Students take courses

Id	Name	Course	CourseID	Department	Building_no	building
1.	Jack	Mathematics	M-1	Math	31	D
4.	Jack	Chemistry	C-1	Chemistry	22	A
2.	Tim	Chemistry	C-1	Chemistry	22	A
3.	Ana	Physics	P-1	Physics	18	S
5.	Ana	Chemistry	C-1	Chemistry	22	A

Third Normal Form (3NF) – continue

very similar to 2NF, but more strict; no functional dependency between non-key attributes at all.

Id	Name	CourseID	Building_no
1.	Jack	M-1	31
4.	Jack	C-1	22
2.	Tim	C-1	22
3.	Ana	P-1	18
5.	Ana	C-1	22

Course ID	Course	Department
M-1	Mathematics	Math
C-1	Chemistry	Chemistry
P-1	Physics	Physics

Building_no	Building
18	S
22	A
31	D

Boyce–Codd Normal Form (BCNF) –

3NF + non dependency between all candidate keys

Example: 3NF, but not BCNF - Today's Court Bookings

Court	Start Time	End Time	Rate Type
1	09:30	10:30	SAVER
1	11:00	12:00	SAVER
1	14:00	15:30	STANDARD
2	10:00	11:30	PREMIUM-B
2	11:30	13:30	PREMIUM-B
2	15:00	16:30	PREMIUM-A

Boyce–Codd Normal Form (BCNF) –

3NF + non dependency between all candidate keys

Example: convert to BCNF - Today's Court Bookings

Rate Type

Rate Type	Court	Member Flag
SAVER	1	Yes
STANDARD	1	No
PREMIUM-A	2	Yes
PREMIUM-B	2	No

Today's Bookings

Member Flag	Court	Start Time	End Time
Yes	1	09:30	10:30
Yes	1	11:00	12:00
No	1	14:00	15:30
No	2	10:00	11:30
No	2	11:30	13:30
Yes	2	15:00	16:30

One More Example – database anormalies

My_Anomly_ex_bank_orig

AccountNo	Name	Address	AccountBal	AccountRate
100001	Lee, Yuan	11 Main Street	23994.58	4.50
100002	Lee, Yuan	11 Main Street	100.74	0.00
100002	Tian, Fu	12 Main Street	100.74	0.00
100003	Wang, Michele	12 Main Street	2500.8	0.00
100004	Dong, Yuan	14 Main Street	32003.98	4.50
100052	Yuan, Ben	16 Main Street	37.38	0.00

One More Example

My_Anomly_ex_bank_orig

AccountNo	Name	Address	AccountBal	AccountRate
100001	Lee, Yuan	11 Main Street	23994.58	4.50
100002	Lee, Yuan	11 Main Street	100.74	0.00
100002	Tian, Fu	12 Main Street	100.74	0.00
100003	Wang, Michele	12 Main Street	2500.8	0.00
100004	Dong, Yuan	14 Main Street	32003.98	4.50
100052	Yuan, Ben	16 Main Street	37.38	0.00

One More Example

My_Anomly_ex_bank_orig

AccountNo	Name	Address	AccountBal	AccountRate
100001	Lee, Yuan	11 Main Street	23994.58	4.50
100002	Lee, Yuan	11 Main Street	100.74	0.00
100002	Tian, Fu	12 Main Street	100.74	0.00
100003	Wang, Michele	12 Main Street	2500.8	0.00
100004	Dong, Yuan	14 Main Street	32003.98	4.50
100052	Yuan, Ben	16 Main Street	37.38	0.00

One More Example

My_Anomly_ex_bank_orig

AccountNo	Name	Address	AccountBal	AccountRate
100001	Lee, Yuan	11 Main Street	23994.58	4.50
100002	Lee, Yuan	11 Main Street	100.74	0.00
100002	Tian, Fu	12 Main Street	100.74	0.00
100003	Wang, Michele	12 Main Street	2500.8	0.00
100004	Dong, Yuan	14 Main Street	32003.98	4.50
100052	Yuan, Ben	16 Main Street	37.38	0.00

One More Example

My_Anomly_ex_bank_orig

AccountNo	Name	Address	AccountBal	AccountRate
100001	Lee, Yuan	11 Main Street	23994.58	4.50
100002	Lee, Yuan	11 Main Street	100.74	0.00
100002	Tian, Fu	12 Main Street	100.74	0.00
100003	Wang, Michele	12 Main Street	2500.8	0.00
100004	Dong, Yuan	14 Main Street	32003.98	4.50
100052	Yuan, Ben	16 Main Street	37.38	0.00
	Gu, Zhen (???)			

Advantages of Normalization

BCNF+ normalization can eliminate all anomalies :

- No Redundancy
- No Inconsistency – all changes can only be made at the same place and keep consistent (because of the key constraints), in DB terminology – get away with all update anormaly.
- Normalization is the process of decomposition, so all the business concepts can be modeled with clear logical relationships
- The entire database system remains consistent over time as the database grows with least redundancy and much durability.
- Strong support to be ACID compliant

Advantages of Normalization

- No(less) data redundancy – means easy management, less storage, etc.
- No headache caused by data operation anomalies. Good for data integrity and consistency.

Disadvantages of Normalization

- Take effort
- May increase complexity in data structure
- Data retrieving efficiency may be discounted due to the need of join of multiple tables; So may not be proper in read-intensive data applications
- Sometimes the constraints may be too strict to be flexible to make some customized change needed.

Disadvantages of Normalization

- Hard to deal with complex data structures such as class, objects, rows in a field.
- Query for comprehensive information can be costly.[6]
- Due to fixed predesigned structure, it is not flexible in terms of restructure of data

Modern applications

- Today companies like Google, Amazon and Facebook deal with loads of data and storing that data in an efficient manner is always a big task. They use NoSQL database which is based on the principles of unnormalized relational model to deal with storage issue.Some of the examples of NoSQL databases are MongoDB, Apache Cassandra and Redis. These databases are more scalable and easy to query with as they do not involve expensive operations like JOIN.

Denormalization

- Normalization and denormalization both have advantages and disavantages. The best practice is always a trade off between the two.
- Denormalization will increase the risk of loss of data integrity and the size of storage, but may gain the simplicity and intuitivity of presenting data.

Denormalization - Example

- Customer (CustomerID, Name, Address, Zip, City, State)

CustomerID [PK]
Name
Address
Zip [FK]

Zip [PK]
City
State

CustomerID [PK]
Name
Address
Zip
City
State

Denormalization - Example

- This is the normalized table design

CustomerID	Name	Address	Zip
101	John	111 Main St	02132
102	Adam	17 Willow St	02451
103	Grace	333 Burke St	02132

Zip	City	State
02132	Boston	MA
02451	Waltham	MA

Denormalization - Example

- This is the denormalized table design

CustomID	Name	Address	Zip	City	State
101	John	111 Main St	02132	Boston	MA
102	Adam	17 Willow St	02451	Waltham	MA
103	Grace	333 Burke St	02132	Boston	MA

Database Operation/Administration

- CRUD (Create/Read/Update/Delete) – four basic operations
- All through SQL (Structured Query Language)
  - Sublanguage
  - DDL (Data definition Language)
  - DQL (Data Query Language)
  - DML (Data Manipulate Language)
  - DCL (Data Control Language)
  - Scope of SQL: Query (select), Manipulate(Insert/update/delete), Definition(Create/Modify tables/columns) , Access Control (permission)

How to Learn SQL

- Same approach as many other languages, get a language reference first
  - https://www.w3schools.com/sql/sql_syntax.asp (Standard SQL, simple)
  - https://docs.oracle.com/cd/B28359_01/server.111/b28286/expressions.htm#SQLRF004 (complete, but more complicated, and may add its own flavor)
- Start from basics (DDL, DML, DQL)
- Extend to more complicate elements (Subquery/Join/Function/Store Procedure/Index/Programming)

Demos and Exercises

- Microsoft Access Templates
- E-R schema in Microsoft Access
- Anomaly
- Data import/export via Microsoft Access (may do some real life demo using actual data file from audience)
- Data import/export via MySQL

Other Stuffs not covered by this tutorial

- Database Administration: concurrency, security, backup, recovery, and many more
- Database Performance tuning: indexing, server configurations
- Database programming
- Database Technology Trend: BigData challenge, Data Warehouse, BI system, NoSQL, Cloud, Hadoop/Spark, etc.

BI Systems

- Reporting System
- Data Mining (Has big overlap with today's ML/AI trend)
- Data Warehouse/Data Mart
- ETL (Extract/Transform/Load)

Big Data

- 4Vs:
  - Volume – how big in storage is need?
  - Variety – how diverse is the data ?
  - Veracity – can data be verified/trusted?
  - Velocity – how fast is the data being generated?

Big Data Technologies

- **Predictive analytics**: to discover, evaluate, optimize, and deploy predictive models by analyzing big data sources to improve business performance or mitigate risk.
- **NoSQL databases**: key-value, document, and graph databases.
- **Search and knowledge discovery**: tools and technologies to support self-service extraction of information and new insights from large repositories of unstructured and structured data that resides in multiple sources such as file systems, databases, streams, APIs, and other platforms and applications.
- **Stream analytics**: software that can filter, aggregate, enrich, and analyze a high throughput of data from multiple disparate live data sources and in any data format.
- **In-memory data fabric**: provides low-latency access and processing of large quantities of data by distributing data across the dynamic random access memory (DRAM), Flash, or SSD of a distributed computer system.
- **Distributed file stores**: a computer network where data is stored on more than one node, often in a replicated fashion, for redundancy and performance.
- **Data virtualization**: a technology that delivers information from various data sources, including big data sources such as Hadoop and distributed data stores in real-time and near-real time.
- **Data integration**: tools for data orchestration across solutions such as Amazon Elastic MapReduce (EMR), Apache Hive, Apache Pig, Apache Spark, MapReduce, Couchbase, Hadoop, and MongoDB.
- **Data preparation**: software that eases the burden of sourcing, shaping, cleansing, and sharing diverse and messy data sets to accelerate data's usefulness for analytics.
- **Data quality**: products that conduct data cleansing and enrichment on large, high-velocity data sets, using parallel operations on distributed data stores and databases.

Summary of Training

- List important points from each lesson.
- Provide resources for more information on subject.
  - List resources on this slide.
  - Provide handouts with additional resource material.