

Lecture 11

Database Management System

Reminder: Practical Exam 2

Next Thursday, 12 Nov, 7pm

Overview

Database Management System

- Motivation
- Properties
- **[OOS]** Conceptual Modeling
- **[OOS]** Logical Modeling
- Simple SQL
 - Data Definition Language (DDL)
 - Data Manipulation Language (DML)
- Interfacing with DBMS

Motivation: Why Database?

Many programs relies on manipulation of large set of data:

- Student database, Point-of-sales system in Supermarket, IVLE, Couremology etc

The data is different, but the operations needed are very similar:

- Create
- Retrieve
- Update
- Delete

Motivation: Why Database?

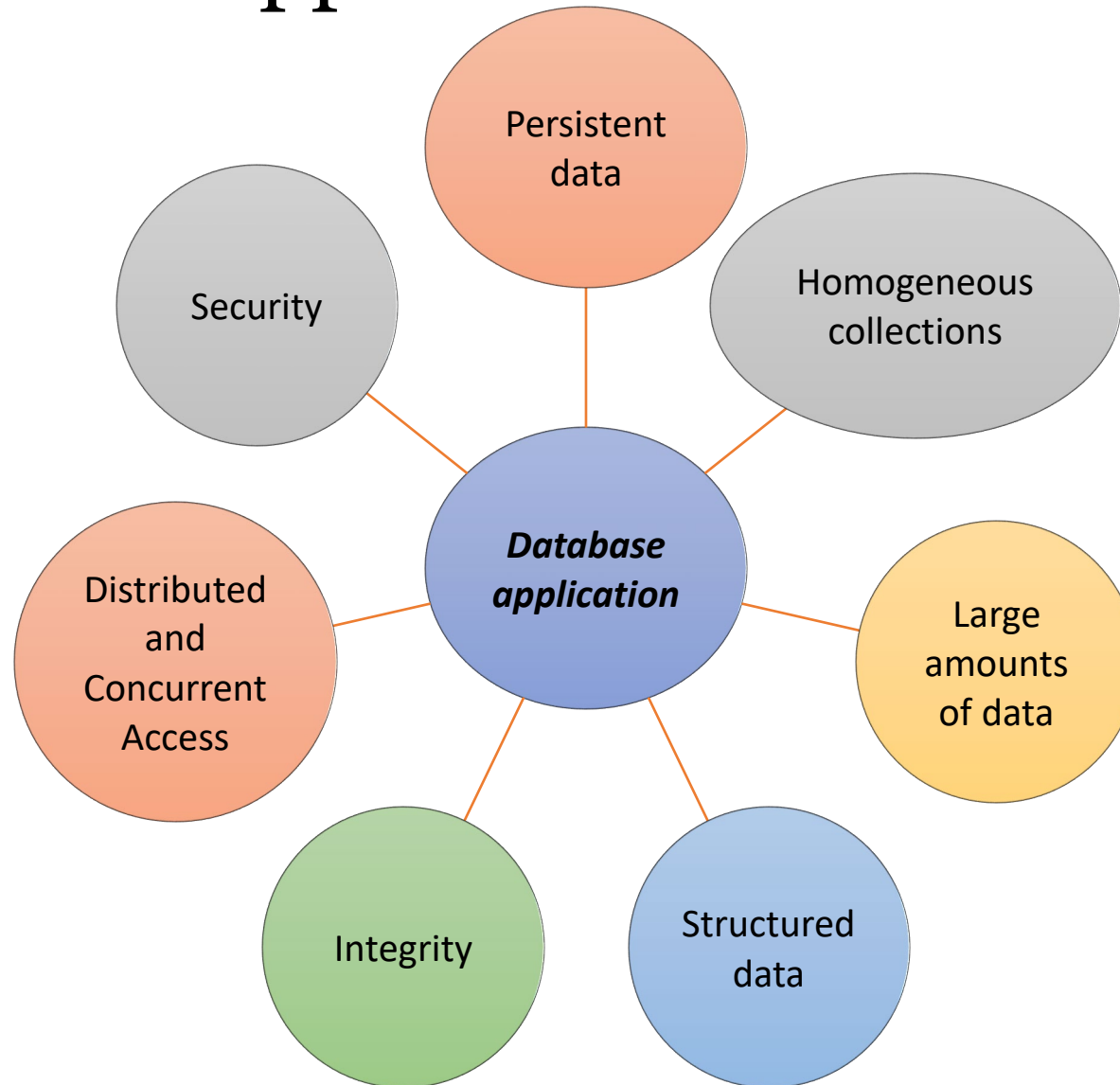
Can we write programs to handle the operations ourselves?

- Yes, but.....

Difficulties:

- Can we reuse the program for **different data set**?
- What if data set is **huge**?
- What if the **system crashes** halfway through an operation?
- What if **many users** want to access the data at the same time?
- Many other tricky requirements.....

Database Applications: Characteristics



Database Management Systems (DBMS)

Generic platforms for the **implementation** and **management** of database applications



ORACLE®

PostgreSQL



Microsoft®
SQL Server®

TURBODB
Embedded



MySQL™

Interacting with DBMS: **Basic Idea**

The DBMS supports **data models**

- User application can be designed around the data **schema**

The DBMS supports **languages** for data **definition** and **manipulation**

- User application can be **programmed** using dedicated languages such as **SQL**

Data Models: Quick Definition

Schema: The framework for defining the general form of the objects and data

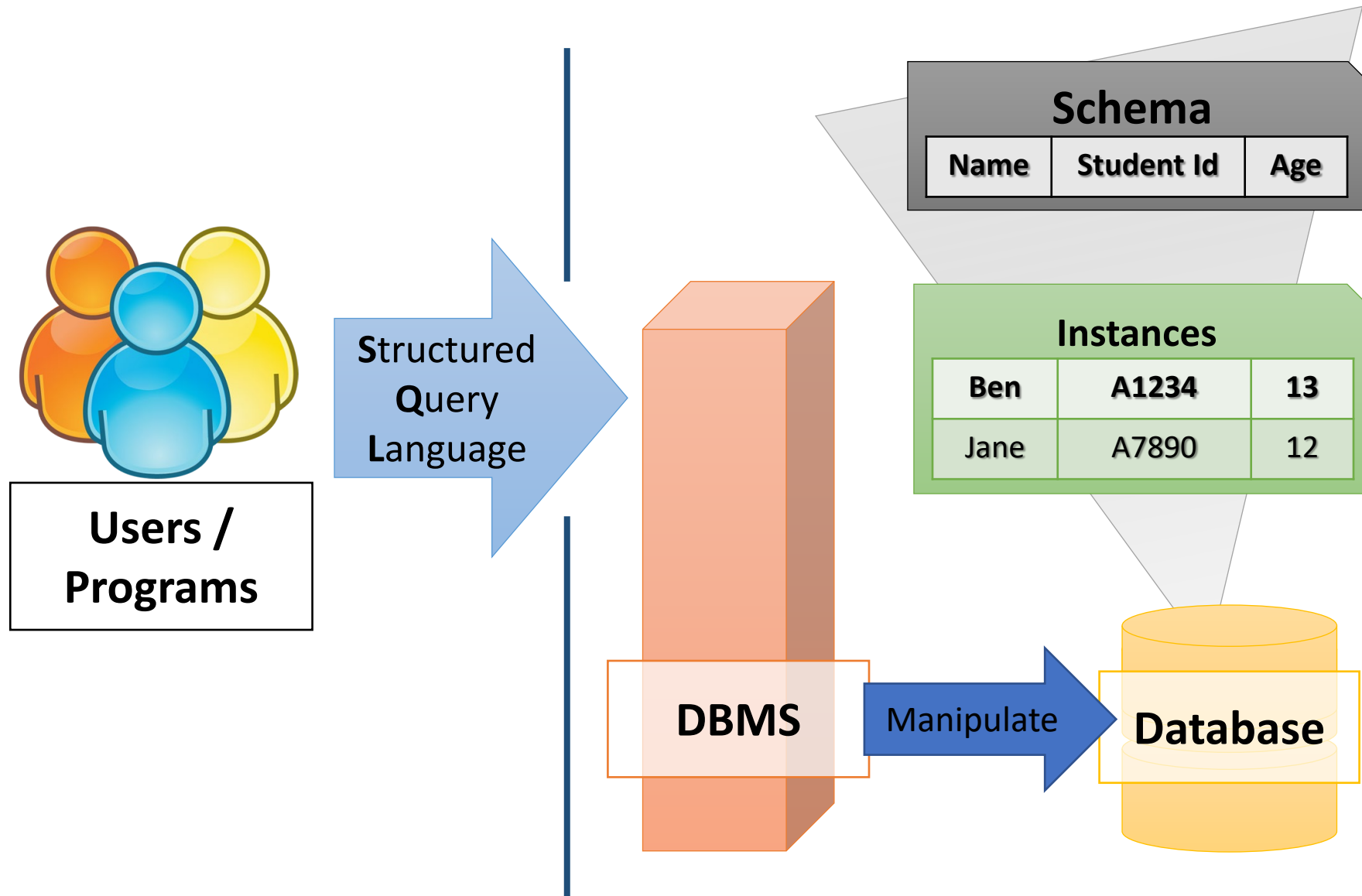
- Analogy: Variables and the datatype of the variables in a C/C++

Instances: At any point in time, the actual content of those objects and data

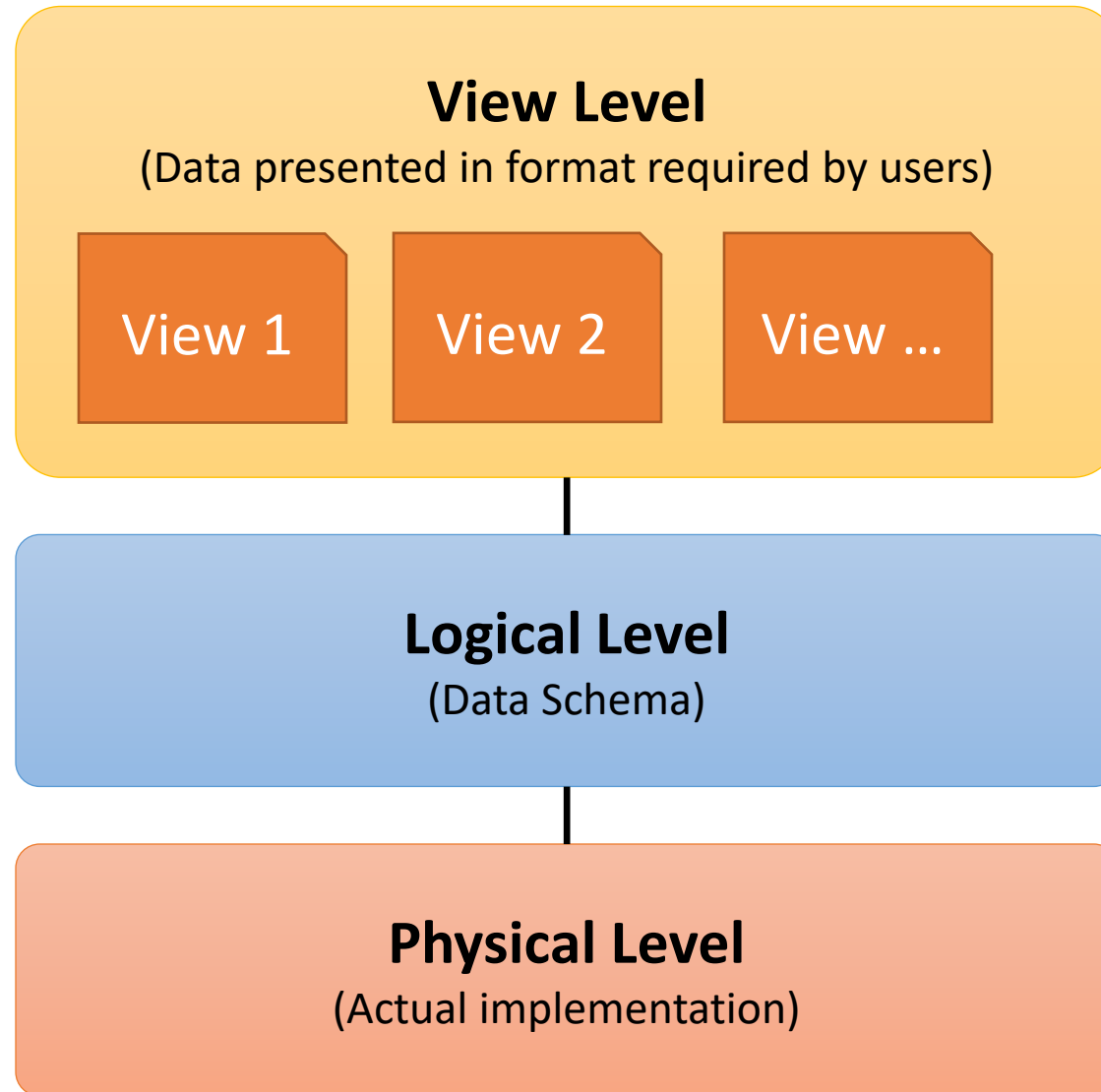
- Analogy: Deciding the what variables and the datatype of those

Schema rarely changes while the instances are often updated

Interacting with DBMS: **Illustration**



DBMS: Layer of Abstractions



Logical and Knowledge Independence

Logical Data Independence

- **Views**: User interactions with the database can ignore the schemas
→ they see what they need in the form they need

Knowledge Independence

- **Complex queries** are available as views
- **Procedures** are stored in the DBMS and the details of implementation hidden
- **Business rules** are captured with integrity constraints and triggers and automatically maintained

Transaction: ACID

A series of SQL statements can be executed as a transaction with guaranteed properties:

Atomicity

- Either ALL or NONE of the statements are executed

Consistency

- Constraints are respected at all times

Isolation

- End result is the same as sequential execution of the statements

Durability

- Execution results can survive permanently

Atomicity

Either ALL or NONE of the statements are executed

Transfer \$X from bank account A into B

- Check account A has $> \$X$
- Deduct \$X from account A
- Increase account B by \$X

What happens if only part of the statements are executed?

Consistency

Constraints are respected at all times

Examples:

- Phone numbers can only be digits (or leading +)
- Credit card numbers have a certain format
- Cheques being cashed must have been issued

Isolation

End result is the same as sequential execution of the statements

- Two concurrent deposits of \$100 into a bank account containing \$500

Transaction 1
Obtain current balance: \$500
Add \$100: \$600
Update balance: \$600

Transaction 2
Obtain current balance: \$500
Add \$100: \$600
Update balance: \$600

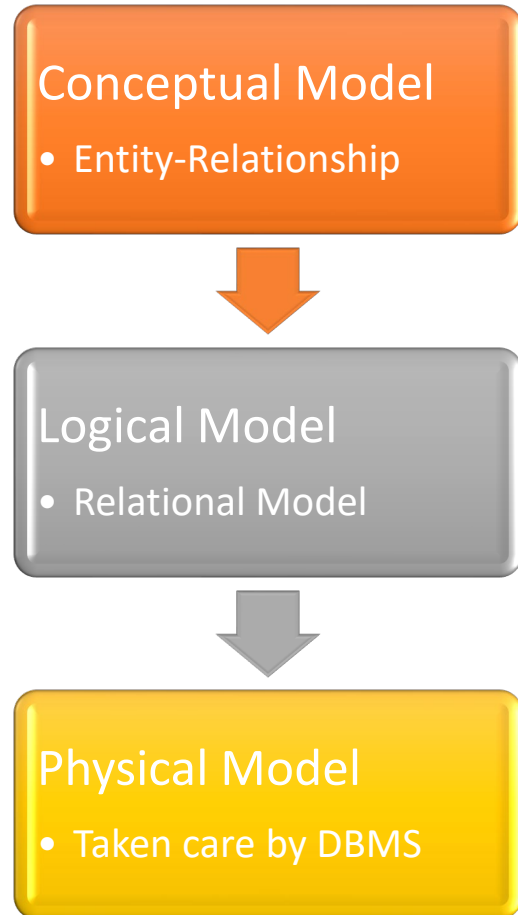
Adding a number on a blackboard

Durability

Execution results can survive permanently

- Bank system crash
- All transaction logs should be backed up

Designing Database Applications



- The models have increasingly more details as we move down
- In actual design, we usually move top-down
- In this lecture, the "logical model" is covered first
 - So that it is clear what we are building towards

[OOS] Logical Modelling

Relational Model

Relation: "**Book**" example

	<i>column</i> <i>domain (or attribute)</i>		<i>relation schema</i> ↓	
	Title	Author	Publisher	ISBN13
	The Future of Learning Institutions in a Digital Age	Cathy N. Davidson, David Theo Goldberg	The MIT Press	978-0262513593
<i>row tuple</i>	Introduction to Algorithms	Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein	The MIT Press	978-0262033848
	The Shallows: What the Internet Is Doing to Our Brains	Nicholas Carr	W. W. Norton & Company	978-0393072228
	Computer Organization and Design	David A. Patterson, John L. Hennessy	Morgan Kaufmann	978-0123744937
<i>table</i> →	Introduction to Algorithms	Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein	The MIT Press	978-0262033848

Terminology: Key Attribute

Key (Candidate Key)

- 1 column or more (**composite key**)
- One key value **uniquely identify** one row in the table
- A table can have more than 1 key
- Primary Key
 - One of the keys will be chosen as the main identifier
 - Used for internal indexing in database
- Foreign Key
 - A value that can be used as the key on another table

Finding the keys: **Try it out!**

Relation: Booking record of LT15

Booking Id	Day	Start	End
76109	Monday	0900	1100
19374	Monday	1500	1700
47129	Friday	1200	1400
40135	Thursday	1830	2030
83920	Wednesday	1200	1500

Relation: Booking record for lecture theatre

Booking Id	LT	Day	Start	End
76109	LT15	Monday	0900	1100
54327	LT19	Monday	0900	1200
35877	LT7A	Friday	1200	1400
40135	LT15	Thursday	1830	2030
42379	LT24	Wednesday	1200	1500

Normalization: Motivation

It is not hard to define a relation

- However, a bad design can result in several problems:
 - **Update Anomaly:** Hard to change the value
 - **Processing Anomaly:** Hard to retrieve information

Normalization: A process to verify the relation(s) define has certain desired properties:

- We will look at First Normalized Form (**1NF**), **2NF** and **3NF**

First Normalized Form: 1NF

What is the problem with the following "Module Enrollment" relation?

Student Id	Student Name	Module 1	Module 2	Module 3
A007007	James Boom	CS1020E	CS2100	EE2020
A123456	Count Lucas	CS2020	EE2020	----
A500500	Uncle Soo	CG3002	----	----
A332789	Aaron Sam	CS1020E	CS2100	----
A991234	Simone Duncan	CS2020	EE2020	PC2020

First Normalized Form: 1NF

Definition:

- each attribute of a table must have atomic (single) values

Student Id	Student Name	Module
A007007	James Boom	CS1020E
A007007	James Boom	CS2100
A007007	James Boom	EE2020
A123456	Count Lucas	CS2020
A123456	Count Lucas	EE2020
.....

Second Normalized Form: **2NF**

What is the problem with the following relation?

Student Id	Student Name	Module
A007007	James Boom	CS1020E
A007007	James Boom	CS2100
A007007	James Boom	EE2020
A123456	Count Lucas	CS2020
A123456	Count Lucas	EE2020
.....

Second Normalized Form: **2NF**

Definition:

- Must be in **1NF**
- Columns should depend on the **whole key**

Student Id	Student Name
A007007	James Boom
A123456	Count Lucas
.....

Student Id	Module
A007007	CS1020E
A007007	CS2100
A007007	EE2020
A123456	CS2020
A123456	EE2020
.....

Third Normalized Form: 3NF

What is the problem with the following relation?

Semester	Module	Lecturer	Office
2016/17 S1	CS1020E	Uncle Soo	COM3-1-23
2017/18 S1	CS2100	Polar Bear	COM1-3-37
2015/16 S2	EE2020	Charmer Chan	COM2-01-1
2017/18 S1	CS2020	Uncle Soo	COM3-1-23
2017/18 S2	EE2020	General Liang	COM2-5-55
.....

Third Normalized Form: **3NF**

Definition:

- Must be in **1NF** and **2NF**
- Column should **directly depend on the key**
 - i.e. no transitive dependencies

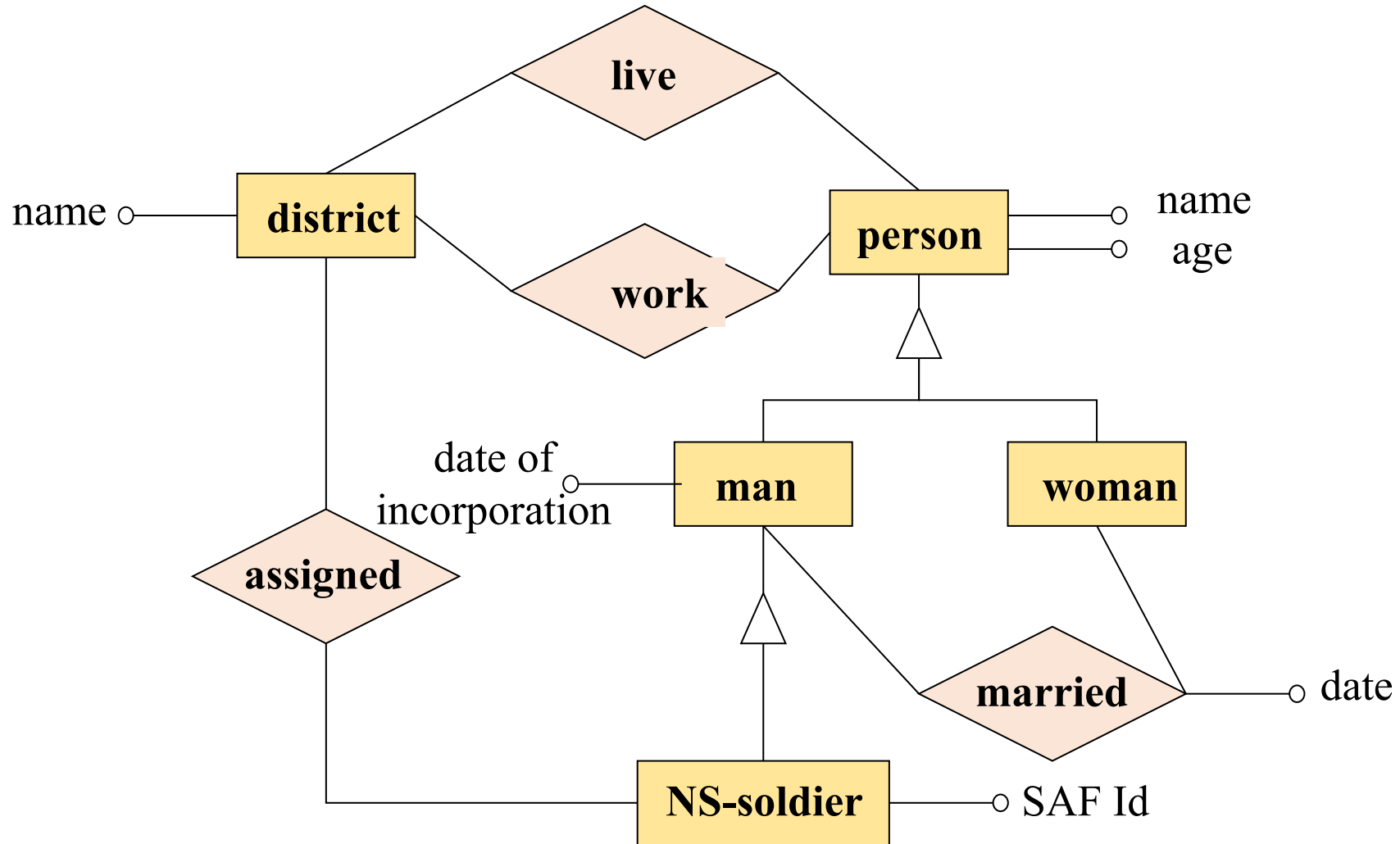
Lecturer	Office
Uncle Soo	COM3-1-23
Polar Bear	COM1-3-37
Charmer Chan	COM2-01-1
General Liang	COM2-5-55
.....

Semester	Module	Lecturer
2016/17 S1	CS1020E	Uncle Soo
2017/18 S1	CS2100	Polar Bear
2015/16 S2	EE2020	Charmer Chan
2017/18 S1	CS2020	Uncle Soo
2017/18 S2	EE2020	General Liang
.....	

[OOS] Conceptual Modelling

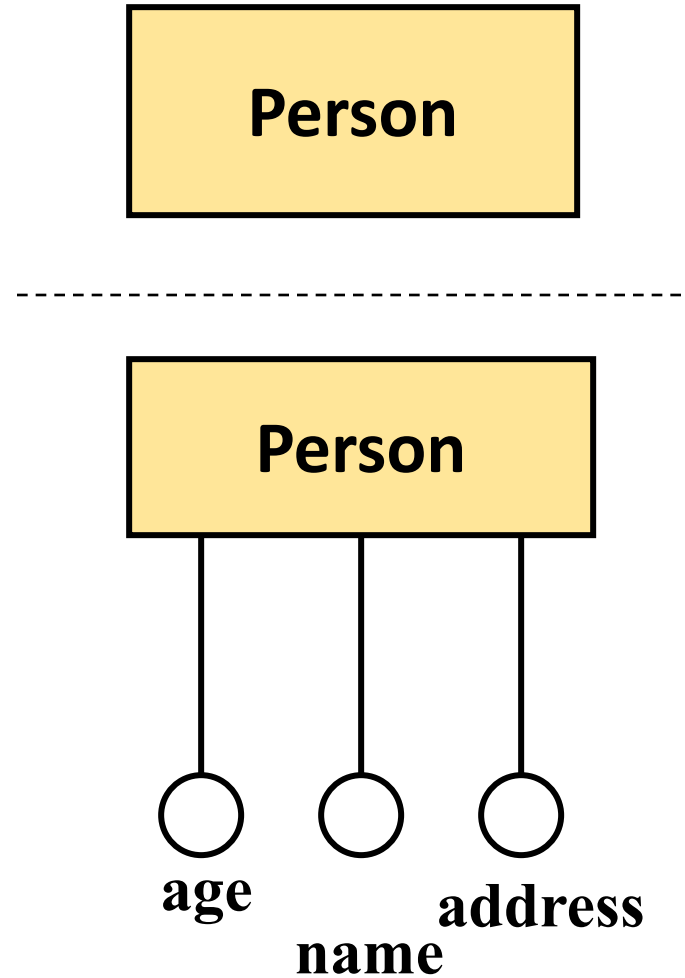
Entity-Relationship Diagrams

ER Diagram: Example



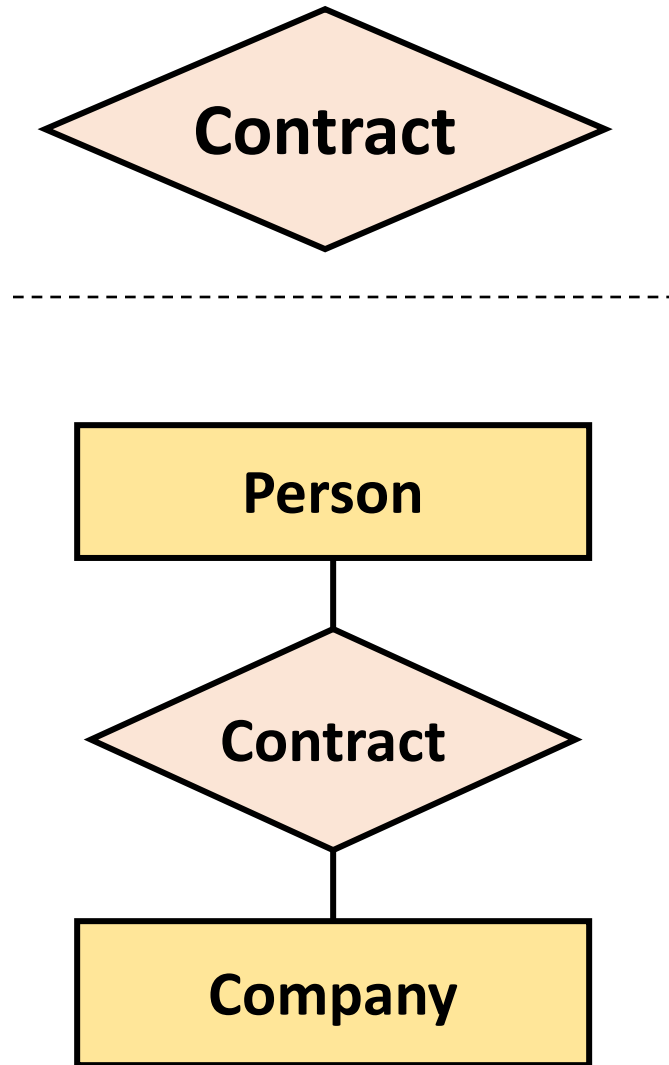
Entities and Attributes

- Entities are identifiable “things”
 - The named box represents a set of entities
- Entities can have attributes
 - All entities in one entity set have the same attributes
 - However the attributes take different values for each entities



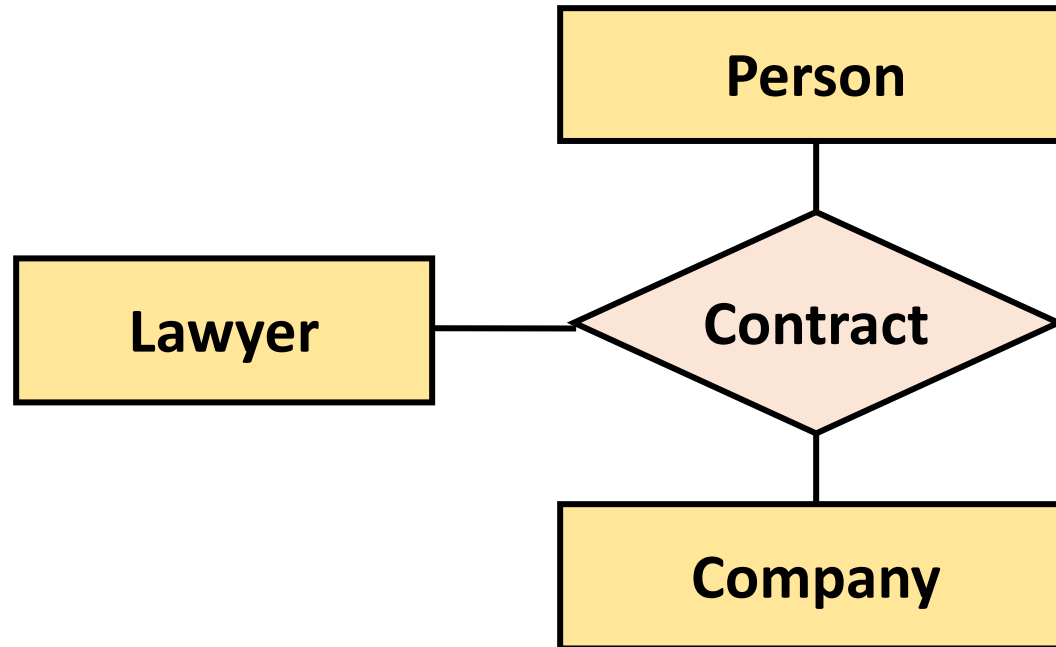
Relationships and Relationship Sets

- Relationships
 - A lozenge represents a set of relationships
 - A relationship associates 2 or more entities
 - A relationship set is a set of relationships associating entities from the same entity sets



N-Ary Relationships

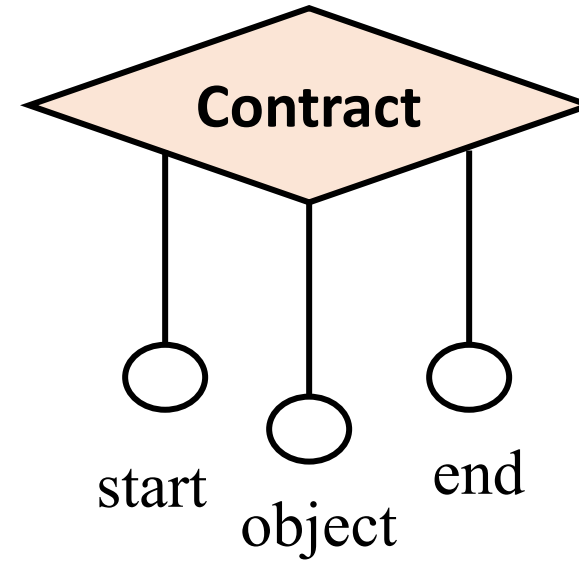
- A relationship can associate more than 2 entities



- We call them n-ary relationships

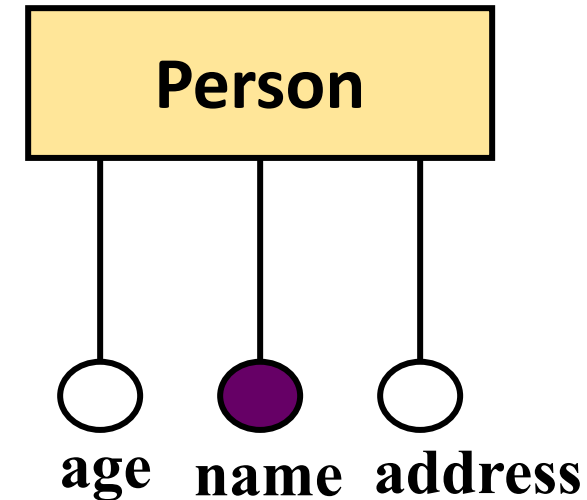
Attributes of Relationships

- Relationship can have attributes
 - All relationships in one relationship set have the same attributes



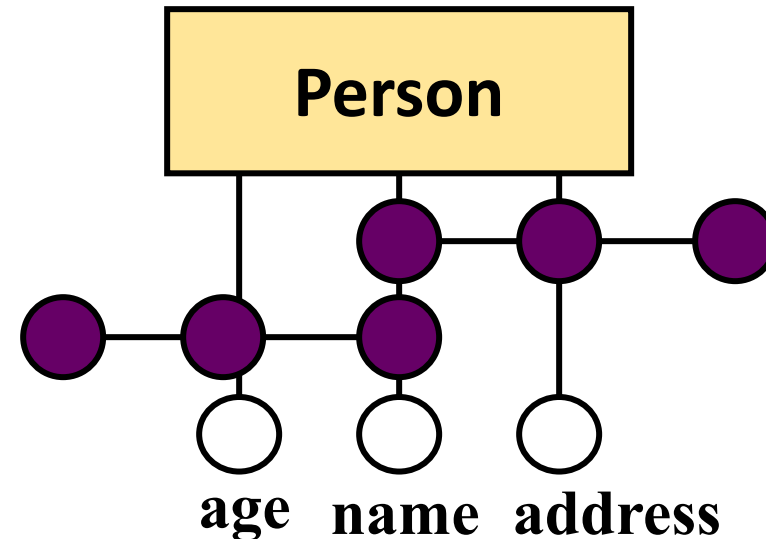
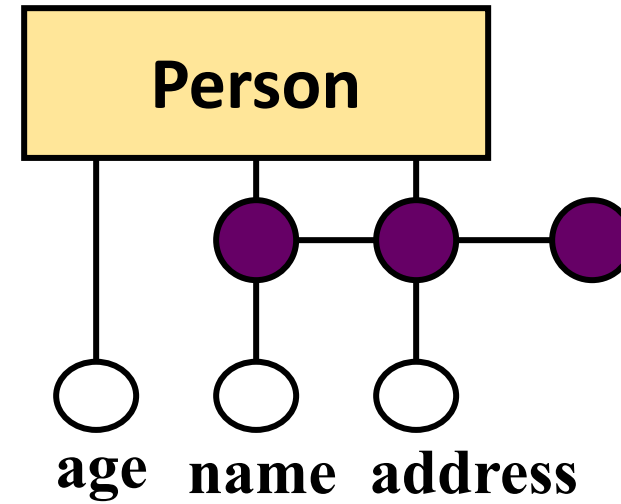
Identity of Entities

- One or more attributes can identify the entity
 - This is a property of all entities in an entity set
 - Notice: at least all attributes identify the entity



Identity of Entities

- A combination of attributes can identify the entity
- Sometimes, there could be multiple combinations to identify an entity



Structured Query Language

How to pronounce SQL? Sqee? Sy-kwee?

Structured Query Language (**SQL**)

Three "sub-languages"

1. Data Definition Language (**DDL**)

- To define relations, views, integrity constraints, triggers

2. Data Manipulation Language (**DML**)

- To update and perform query

3. Database Control Language (**DCL**)

- To define access rights, concurrency control, etc....

User View of Database

Data in database is organized as tables

- Relationship can be defined between tables

Column Name	Column			
	title	authors	pages	prices
One Record	DBMS for Dummy	Bruce Benner	213	79.90
	D for Database	Natasha	197	50.75
	ABC of Database	James Xavier	78	10.99

Example: “**book**” table

SQL DDL – Create Table

SYNTAX	<pre>CREATE TABLE <table_name> (<column_name> <datatype> [constraint] [, <column_name> <datatype> [constraint]])</pre>
EXAMPLE	<pre>CREATE TABLE book(title VARCHAR(256), authors VARCHAR(256), pages INT, price FLOAT)</pre>

Caution: Datatype definition is DBMS dependent!

SQL

Data Manipulation Language

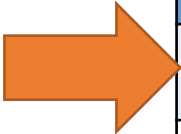
SQL DML: Insertion example

SYNTAX

```
INSERT INTO <table_name> [COLUMNS]
VALUES ( <values for 1 row> )
```

EXAMPLE

```
INSERT INTO book
VALUES ( "DBMS for Dummy",
        "Bruce Benner", 213, 79.90 )
```



title	authors	pages	prices
DBMS for Dummy	Bruce Benner	213	79.90
.....

SQL DML: Query

SYNTAX	SELECT [DISTINCT] <i><target-list></i> FROM <i><relation-list></i> [WHERE <i>qualification</i>]
---------------	--

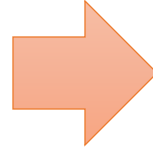
EXAMPLE	SELECT * FROM <i>book</i>
----------------	--

title	authors	pages	prices
DBMS for Dummy	Bruce Benner	213	79.90
D for Database	Natasha Romanov	197	50.75
ABC of Database	James Xavier	78	10.99
.....

SQL DML: Query [Cont]

EXAMPLE

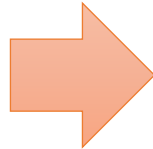
```
SELECT title, pages
FROM book
```



title	pages
DBMS for Dummy	213
ABC of Database	78
D for Database	197
.....	...

EXAMPLE

```
SELECT title, pages
FROM book
WHERE pages > 200
```



title	pages
DBMS for Dummy	213
.....	...

EXAMPLE

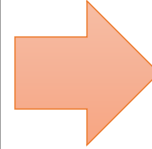
```
SELECT title, price
FROM book
WHERE pages < 200
AND price < 50
```



title	price
ABC of Database	10.99
.....	...

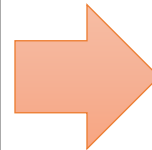
SQL DML: Query [Cont 2]

EXAMPLE	SELECT title, pages FROM <i>book</i> ORDER BY title ASC



title	pages
ABC of Database	78
D for Database	197
.....	...

EXAMPLE	SELECT title, pages FROM <i>book</i> ORDER BY pages DESC



title	pages
DBMS for Dummy	213
D for Database	197
ABC of Database	78
.....	...

EXAMPLE	SELECT <i>title, price</i> FROM <i>book</i> WHERE price < 50 ORDER BY title ASC



title	price
ABC of Database	10.99
.....	...

SQL DML: More about Query

Common Operators	Common Relationship
= (equal)	AND
<> (not equal)	OR
<	NOT
>	
<=	
>=	
BETWEEN	
LIKE	
IN	

Note: Those in red are beyond the scope of this course

[OOS] SQL DML: **Update** example

You can modify (update) any field of existing records by the "**UPDATE**" clause

SYNTAX

```
UPDATE <table_name>  
  SET col1=value1 <, col2=value2, ...>  
  <WHERE criterial <,criteria2, ...> >
```

EXAMPLE

```
UPDATE book  
SET price = 29.90  
WHERE title='Database for Dummy';
```

[OOS] SQL DML: **Deletion** example

You can remove existing records by the '**DELETE**' clause

SYNTAX

```
DELETE FROM <table_name>  
<WHERE criterial <,criteria2, ...> >
```

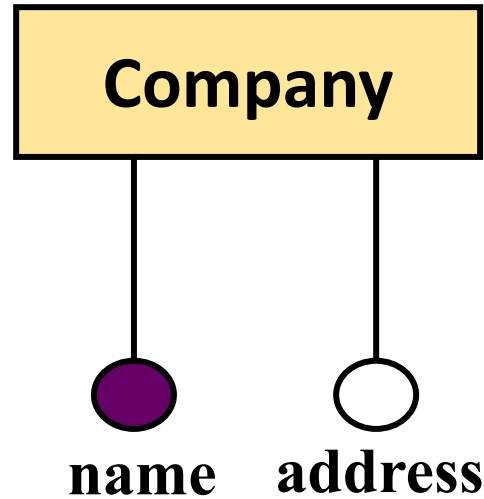
EXAMPLE

```
DELETE FROM book  
WHERE title LIKE '%Data%'
```

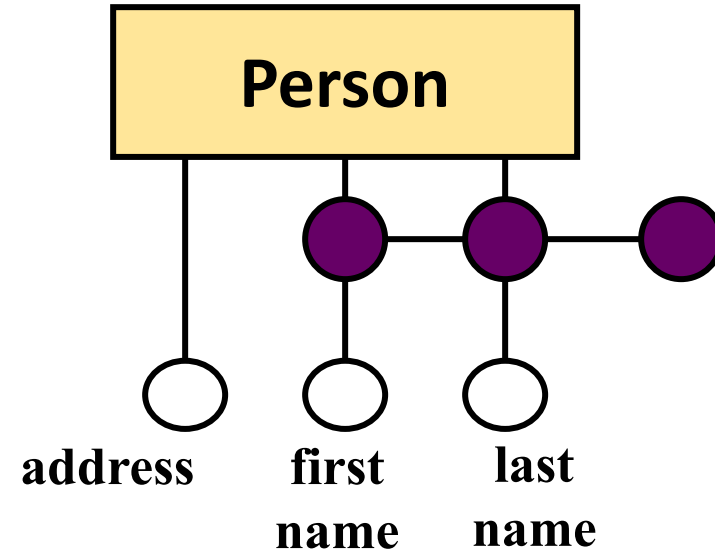
ER Diagram to DB Schema

Conceptual → Logical

Creating Table from Entities Set

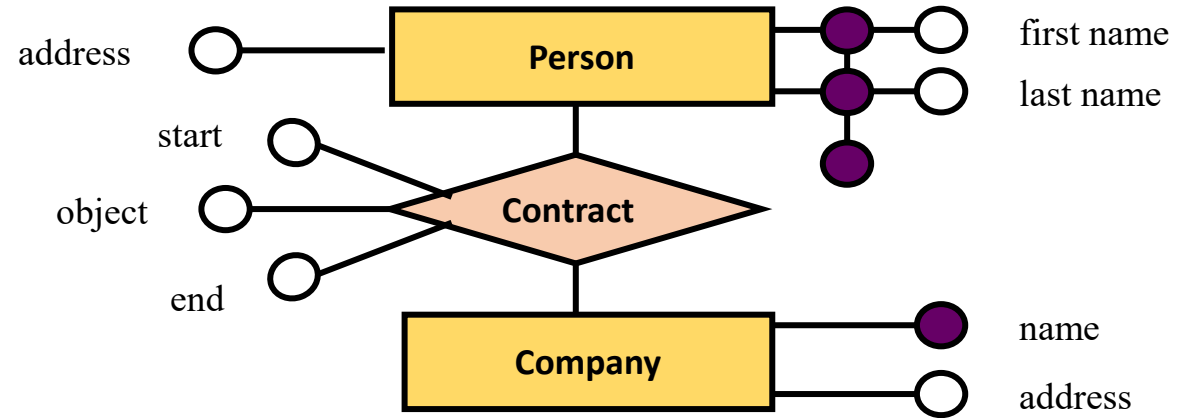


```
CREATE TABLE company (  
  name VARCHAR(64) PRIMARY KEY,  
  address VARCHAR(128) )
```



```
CREATE TABLE person (  
  first_name VARCHAR(32),  
  last_name VARCHAR(32),  
  address VARCHAR(128),  
  PRIMARY KEY (first_name,  
               last_name) )
```

Creating Table from Relationship Sets



```
CREATE TABLE contract (  
  start DATE,  
  end DATE,  
  object VARCHAR(128),  
  pfirst_name VARCHAR(32),  
  plast_name VARCHAR(32),  
  cname VARCHAR(64),  
  PRIMARY KEY (pfirst_name, plast_name, cname),  
  FOREIGN KEY (pfirst_name , plast_name ) REFERENCES person(first_name,  
  last_name),  
  FOREIGN KEY (cname ) REFERENCES company(name)  
)
```

SQL: More!

For your own exploration
(i.e. not in the scope of this course)

SQL: Hmm... not very powerful?

We only scratched the surface 😊

SQL can execute very complex query:

- Link several tables together
- Ranking
- Remove duplicates
- Accumulate / generate result on the fly
- etc

Quick Example:

- Tables: Students, Courses, Degrees
- Queries: All courses with at least one students, students taking more than 3 courses, Students fulfilled requirement for a degree, etc

Where to learn more?

Good internet websites:

- <http://www.w3schools.com/sql/default.asp>
 - Good for PHP too
- <http://sqlfiddle.com>
 - Online SQL "playground"
- <http://sqlzoo.net/w/index.php>

Interfacing with DBMS

How do we interface with DBMS

Interfacing with DBMS

High level programming languages

- **C/C++**: SQLAPI++, MySQL connector, etc
- **Python**: Many SQL interfaces for different DBMS
- etc...

Many web pages are built on a DBMS:

- **LAMP** (Linux + Apache + MySql + PHP)
- **WCMS** (Web-based Content Management System), e.g. Wordpress
- etc....

References and Acknowledgement

“Introduction to database systems”, by S.Bressan and B. Catania, McGraw Hill publisher

Some slides adapted from CS2102: Introduction to Database, by A/P. Stephane Bressan

END