Dr Greg Wadley

INFO 90002
Database Systems & Information Modelling

Week 02

Designing and Implementing a Database

Structure of today's lecture

- first hour: Designing Databases
  - homework: noun-verb analysis of 'Enviro-R-Us' case
  - the database life-cycle
  - modelling a database for a department store
    - conceptual model
    - logical model
    - physical model
- second hour: Implementing and Using Databases
  - create the database and tables
  - populate tables with data
  - query data
  - change data

Noun-verb analysis (homework)

- The client (Environments'R'Us Pty Ltd) is building a new web site and wants a backend database system to underpin their business activities.
- Their business model involves sourcing as much content as possible for little cost, by providing a flexible, efficient structure and a process for other people (environmental experts, academics, opinion leaders), to publish articles about the environment on the site. The idea is to attract as many site visits as possible from the global web-surfing population, and convert some readers to Members who will pay money for services provided by the website.
- The articles will be reviewed and edited by the Environments'R'Us *Editor* before they go live (are published) on the web site.
- Anybody can forward an article for inclusion to the site at no cost to them, but only after they have registered their name and email address with Environments'R'Us. Once they have registered, they are considered to be a *Member* of the site, and may receive email from Environments'R'Us Pty Ltd.

Noun-verb analysis

- In addition to publishing articles on the website, the company maintains a list of forthcoming *Events* that should be of interest to people concerned with the environment – seminars, courses, conferences, meetings, festivals. Events and their descriptions (title, type, description, date, durations, unit of duration (hours, days, weeks), cost), are sourced from members. However, a member must become an *Affiliate Member* before they can publish an Event. Publishing an event will have a cost associated with it: a cost per day, up until the event starting date arrives.
- Affiliate Members must register more information about themselves than Members, ie: full-name, address, title, gender, biosketch, phone number, photo (optional). In addition, an Affiliate Member must be a representative of an Organisation. They must also supply information about the Organisation: name, full address, url of organisation's web site. They must indicate whether billing information should be addressed to the individual contact (the affiliate member), or the Organisation. The Affiliate Member's organisation details will be listed on the www.environment.biz web site, free of charge.

Noun-verb analysis

- In addition to articles and events, there is to be a listing of environmentally-friendly *Products* which Affiliate Members can put up on the site. A listing in the Product area of the web site has an associated cost to Affiliate Members: cost-per-day once the Product description is published. The Affiliate Member can upload the description of the product, but the publishing of the description is actioned by the Account Manager within the Environment'IS'Us company.
- In addition, Affiliate Members can put up descriptions of *Services* which have some orientation to environment issues and concerns – for example an architectural service which gives advice about the energy usage of a building. The Affiliate Member can upload the description of the Service, but the publishing on the web site of the description is actioned by the Account Manager within Environment'IS'Us.

Noun-verb analysis

- The system will include basic money transaction details: an *E_Invoice*, records of *Payment* and an *E_Receipt* which acknowledges payments made. An E_Invoice (called a Bill by many people) is to be generated from the database monthly, incorporating the charges accumulated that month. An E_Invoice will be an email sent to the member indicating how much money they currently owe. The *E_Receipt* will be an emailed receipt, acknowledging a payment that the member recently made. There will also be an *E_Reminder* statement – an email sent monthly, to those members who haven't yet paid a due account (a due account, is any account that hasn't been fully paid within 30 days since an E_Invoice was sent requesting payment). E_Invoice, Payments, E_Receipt and E_Reminder details need to be stored in the system, long after the monies have been paid - for forwarding into the client's offline account system (we are not concerned with the accounting system, though we acknowledge that some information here will be duplicated in the conventional accounting system).

Noun-verb analysis

- All articles, events, products and services have fields recording Country and City.
- The client needs to keep track of system events such as: when a pending article gets published (*goes live*) by the Editor; when did a regular Member became an Affiliate Member.
- From the point of view of the ‘customers’ of Environments’R’Us – ie. the casual web-surfers who chance across the site, or come to the site regularly to read material – all of the things on the site are seen as *resources* – an article can be seen as a resource, as can be a forthcoming event, as can be an environmentally-friendly Product, as can be an environmentally-oriented Service.
- In addition to the above information, the client has supplied some rudimentary information about data fields they would like in the system, but they are open to suggestion about what should and what shouldn’t be included. The rudimentary data items include:

Noun-verb analysis

- *Article* information should include: title, abstract, authors, category, keywords, copyright notice.
- *Courses* and similar events should include: title, description, contact person, keywords, category (short, certificate, university degree, etc), location, url address of further course information.
- *Product* information should include: name, description, sales person, category (book, instrument, software, other), keywords, location, thumbnail image, retail cost, url address for further product information.
- *Services* information should include: name, description, consultant, category (consulting, instructional, other).

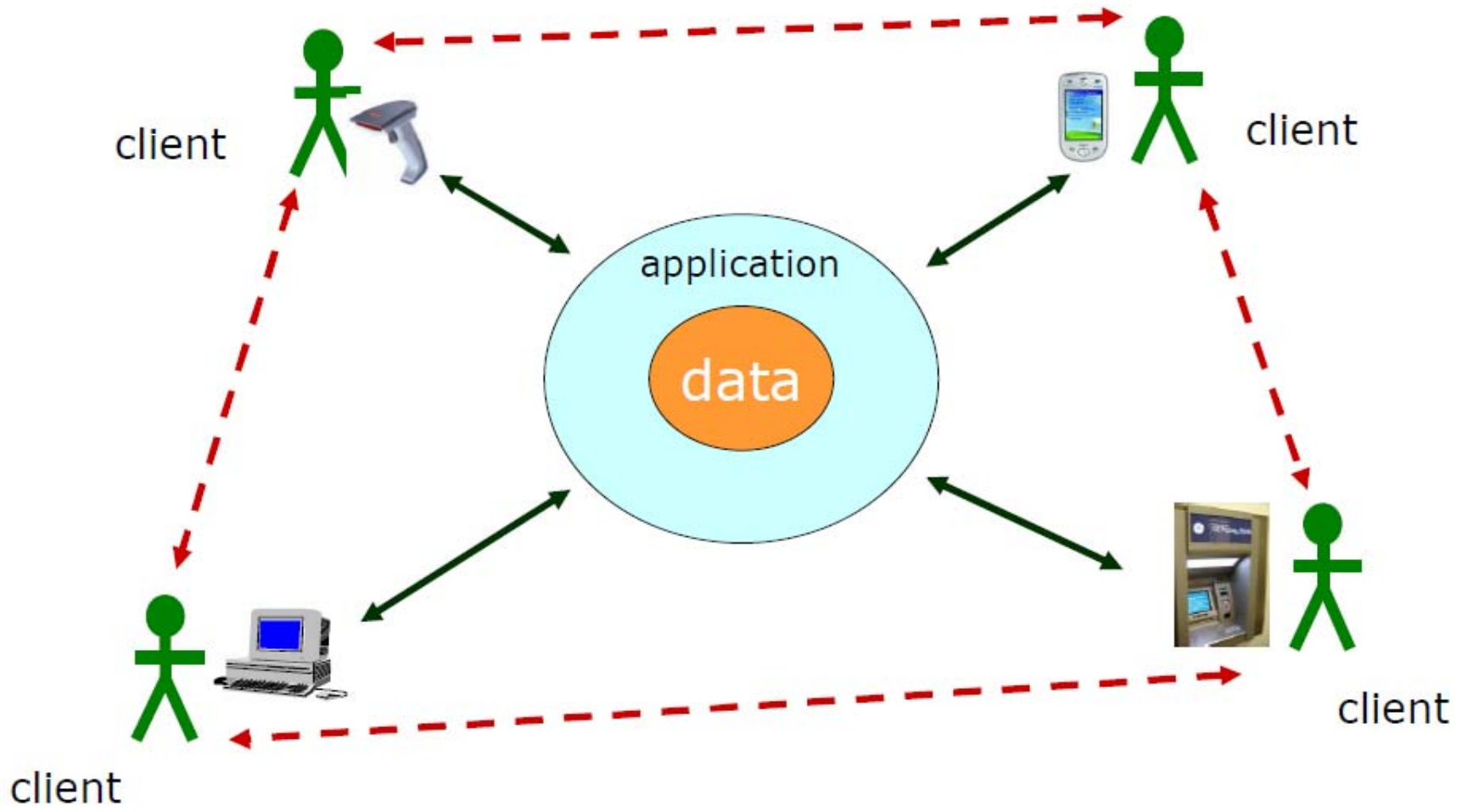
Noun-verb analysis

Noun Verb Analysis for Environment'IS'Us Pty Ltd

Nouns		**Verbs**	
Potential Entities	*Potential Attributes*	*Potential Relationships*	*Business Rule*
article	title	receive	Anyone can register as a member by forwarding name and email.
	abstract	review	A member can forward an article for inclusion to site.
	author	edit	A member receives email from Environment'IS' Us P/L.
	category	publish	An editor reviews articles.
	keyword	supply	An editor edits articles.
	copyright_notice	...	A member can become an affiliate_member by supplying more data.
editor	name		An affiliate_member must represent an organisation.
member	email_address		Affiliate_members must supply organisation data.
	name		Affiliate_members must nominate a billing-address.
			Environment'IS'us will list a member's organisation on the web-site.
affiliate_member	full_name	register	An affiliate_member can publish an event.
	address	represent	An affiliate_member can upload product details.
	biosketch	upload	An account_manager can publish product details.
	phone_no		Publishing an event has a cost-per-day.
	photo		Cost is accumulated per-day until an Events starting date.
	billing_address		Publishing a product has a cost-per-day.
listing			An affiliate_member can upload description of a service
event	type (seminar, course, conference, meeting, festival)		An account_manager can publish service descriptions.
			Account generates E_Invoices each month.
	title		Account sends an E_Receipt on receiving a payment.
	description		Account emails E_Reminder monthly, if an outstanding amount.
	date		...
	duration		
	unit_of_duration		
	cost_per_day		
organisation	name		
	address		
	url		
product	name		
	description		
	sales_person		
	keywords (multi-valued)		
	location		
	thumbnail		
	retail_cost		
	url		
account_manager	name	list	
		publish	
service		description	
Account	amount_owed		
	amount_paid		
E_Invoice	amount_due		
E_Reminder	amount_outstanding		
E_Receipt	amount_paid		
...	...		

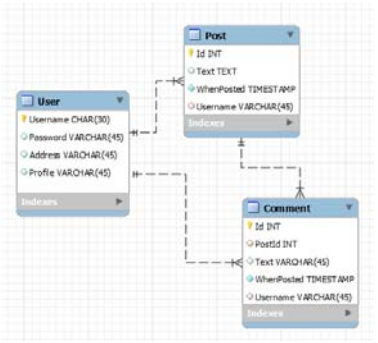
Database Development Lifecycle

System architecture



Recap week 1: Database lifecycle

- Design the database
  - data modelling, E-R diagrams



- Implement the database
  - data definition language (DDL)

•Create
•Drop
•Alter
•Rename

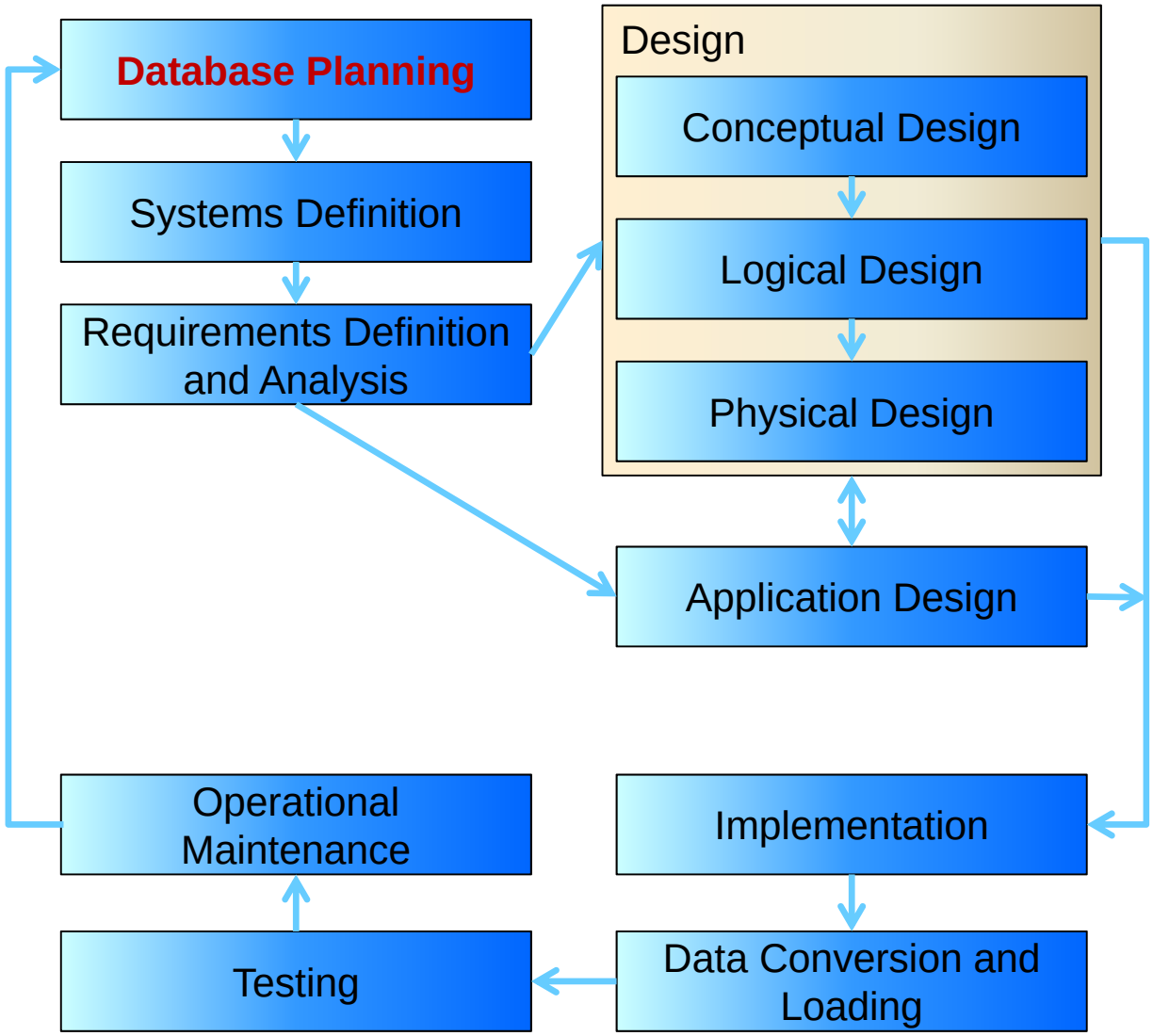
- Data access / programming
  - data manipulation language (DML)

•Select
•Insert
•Update
•Delete

- Database administration
  - data control language (DCL)

•Grant
•Revoke

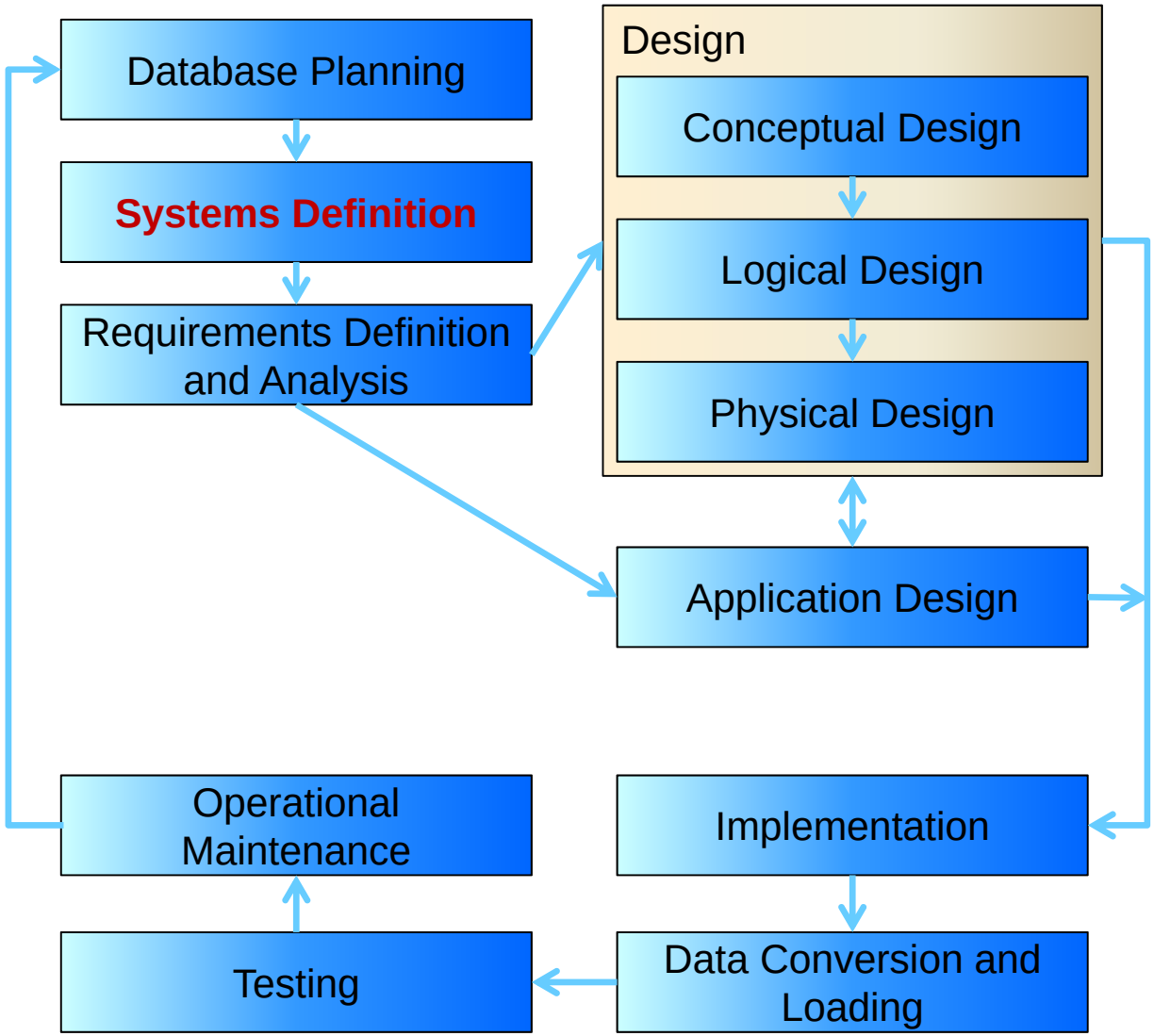
Database Development Lifecycle



- Planning how to do the project.
  - How does the enterprise work
  - Enterprise data model
- How can the stages be completed efficiently and effectively.

- Outside scope of the course

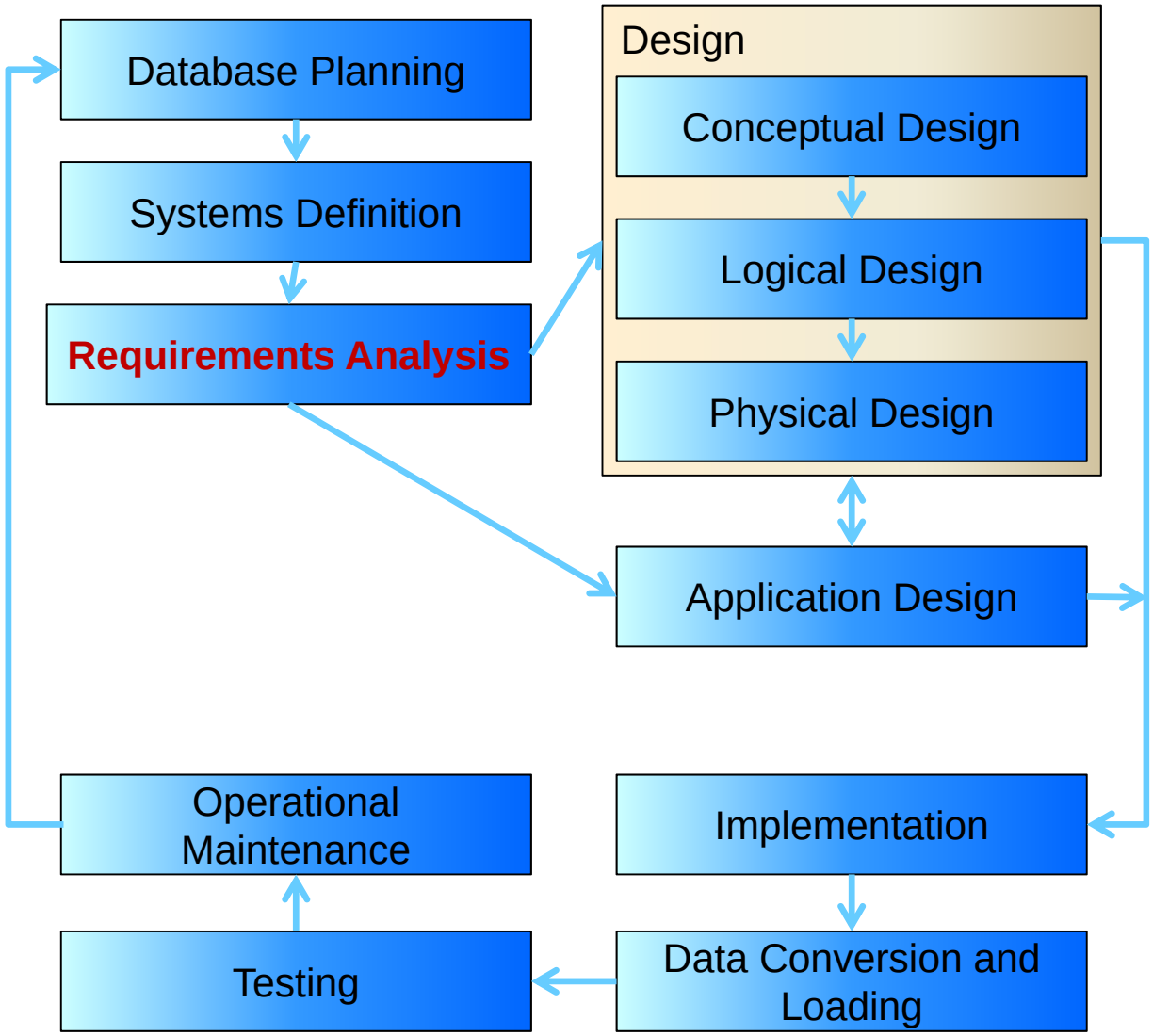
Database Development Lifecycle



- Specifying scope and boundaries
  - Users
  - Major user views
  - Application areas
- How does it interact with other systems
- User views – how the system operates from differing perspectives

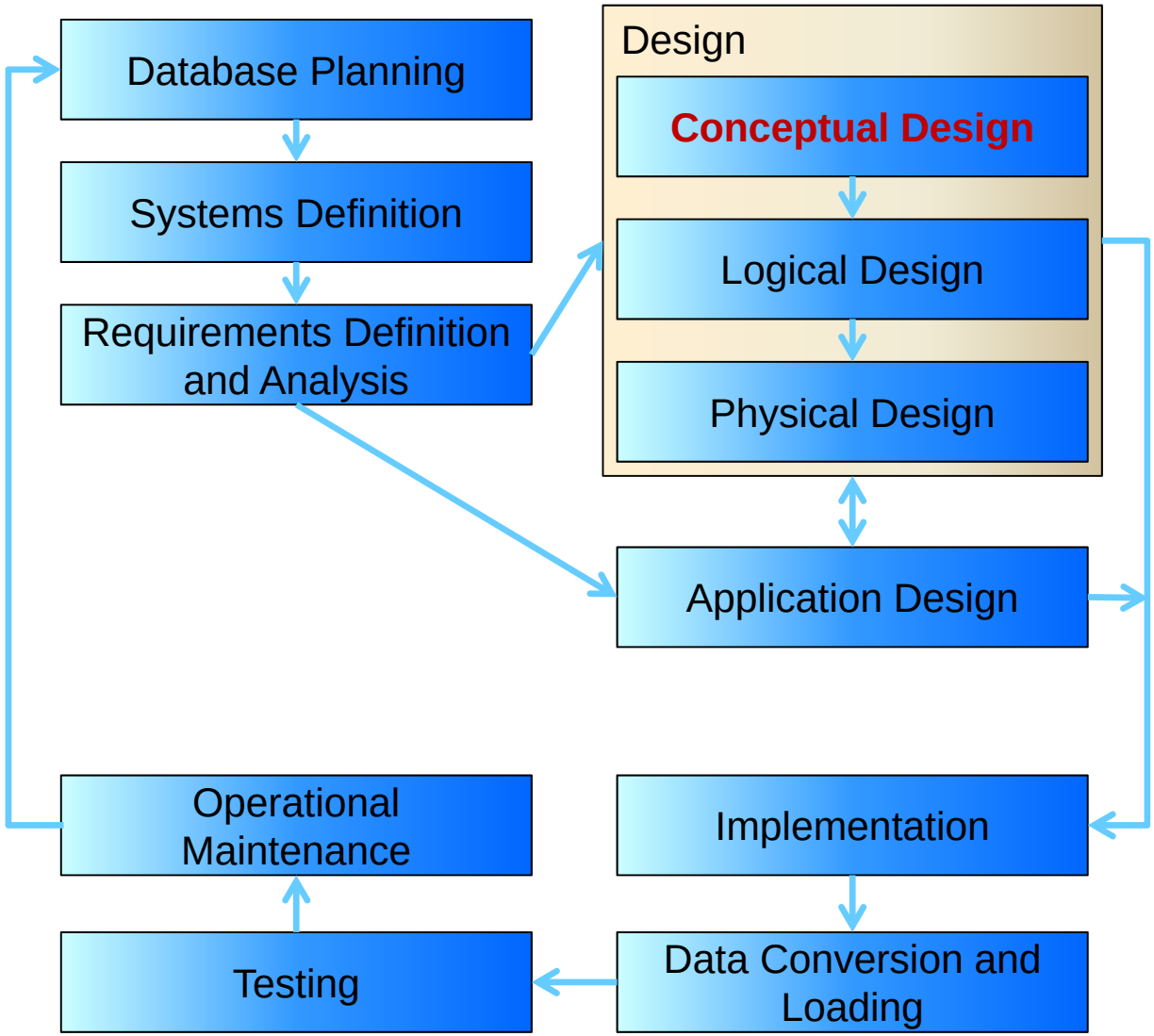
- Outside scope of the course (slightly)

Database Development Lifecycle



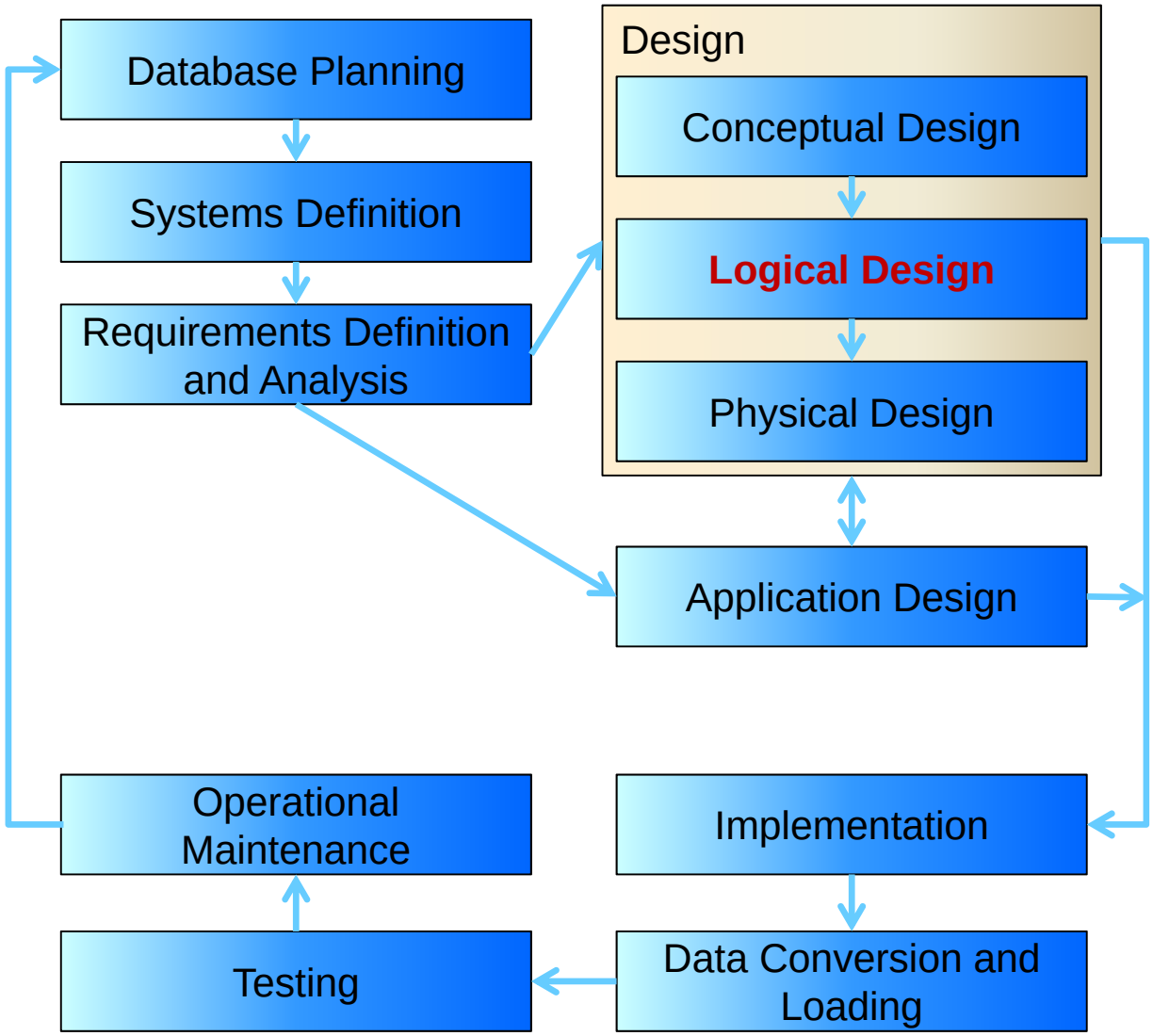
- Collection and analysis of requirements for the new system

Database Development Lifecycle



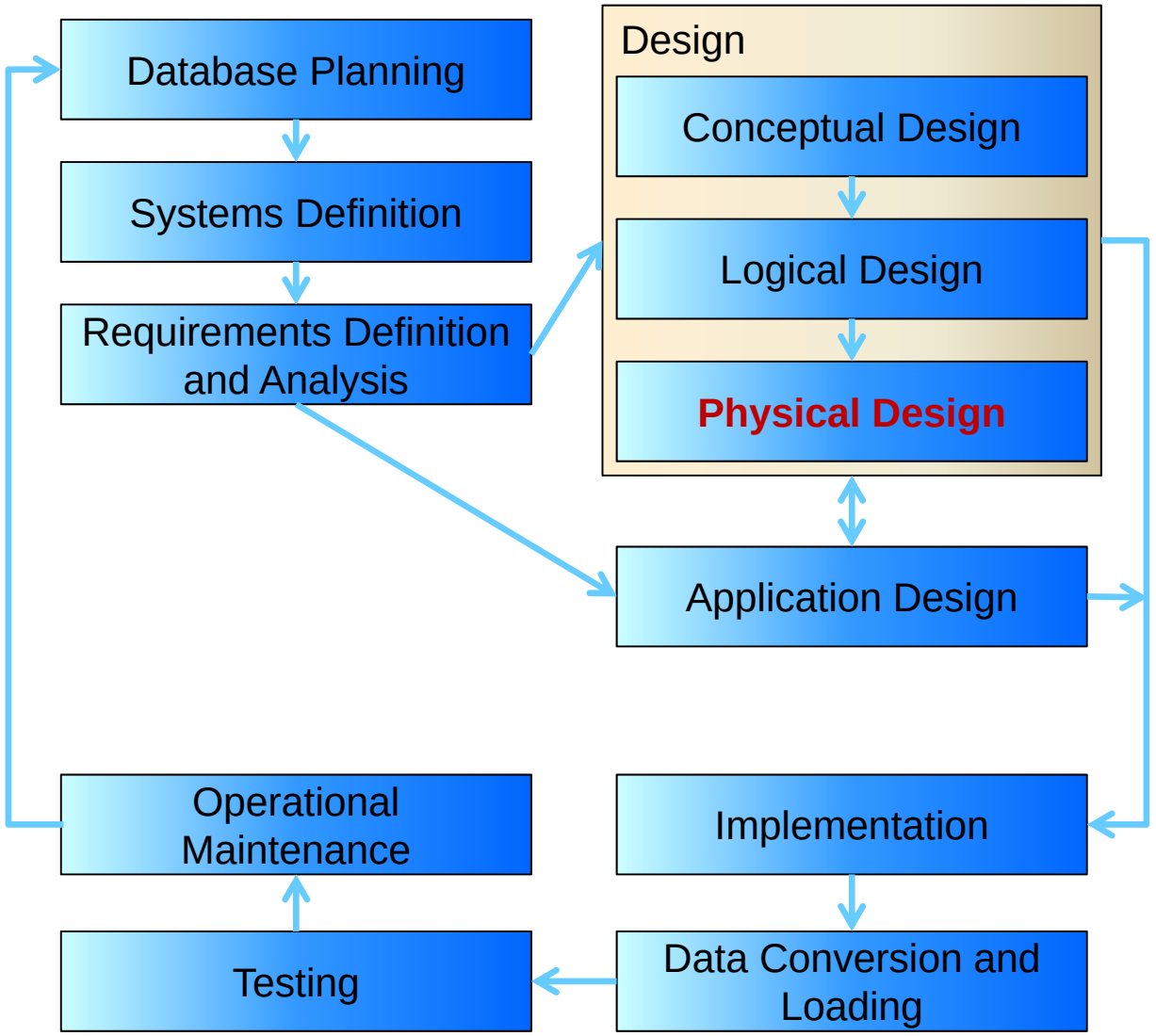
- High-level, first-pass model of entities and their connections
- Typically omits attributes
- Could potentially be implemented in a non-relational database
- Thus can include many-to-many relationships, repeating groups, composite attributes

Database Development Lifecycle



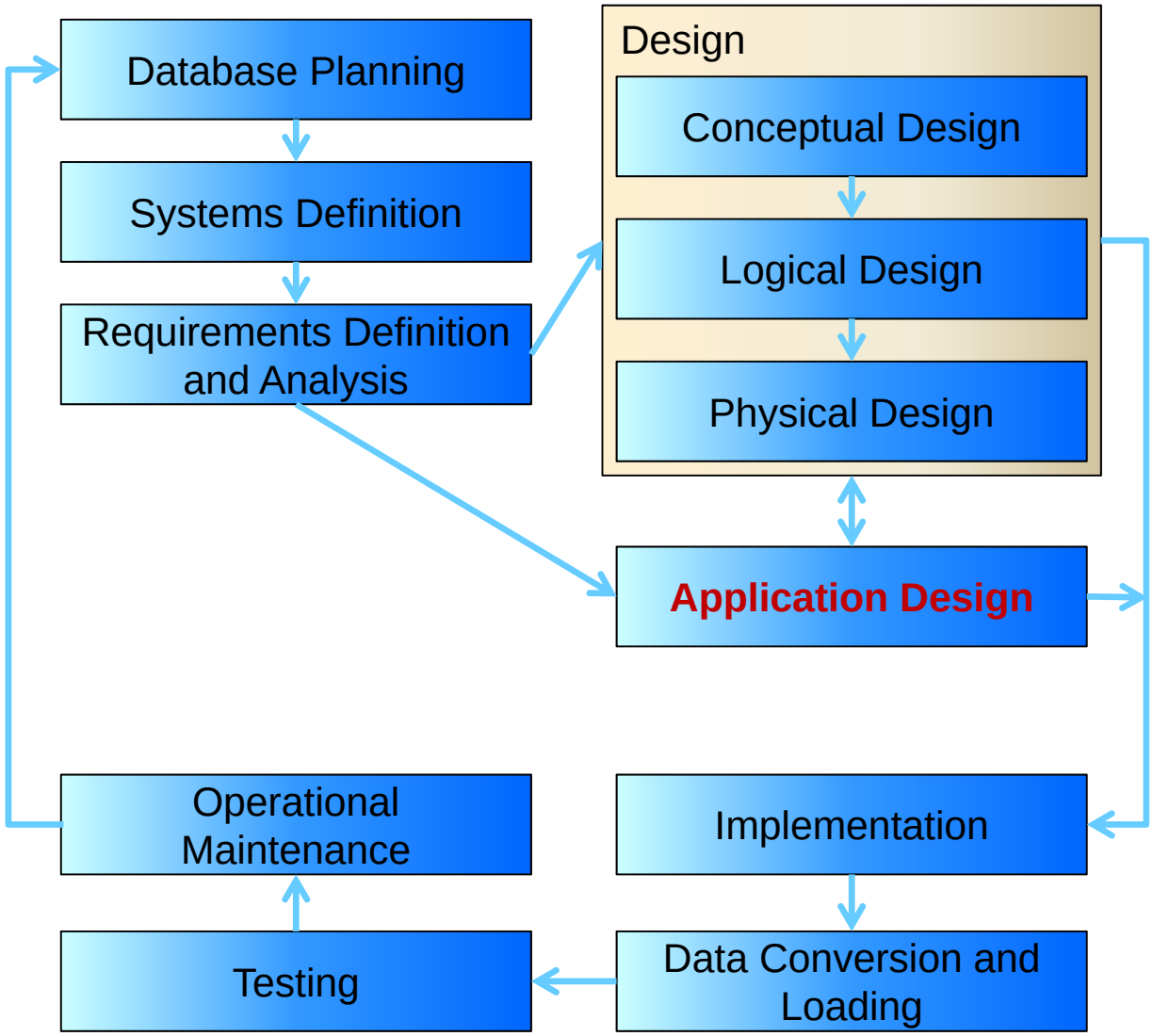
- Builds on the conceptual design
- Designing now for a relational database
- Includes columns and keys
- Independent of a specific vendor and other physical considerations

Database Development Lifecycle



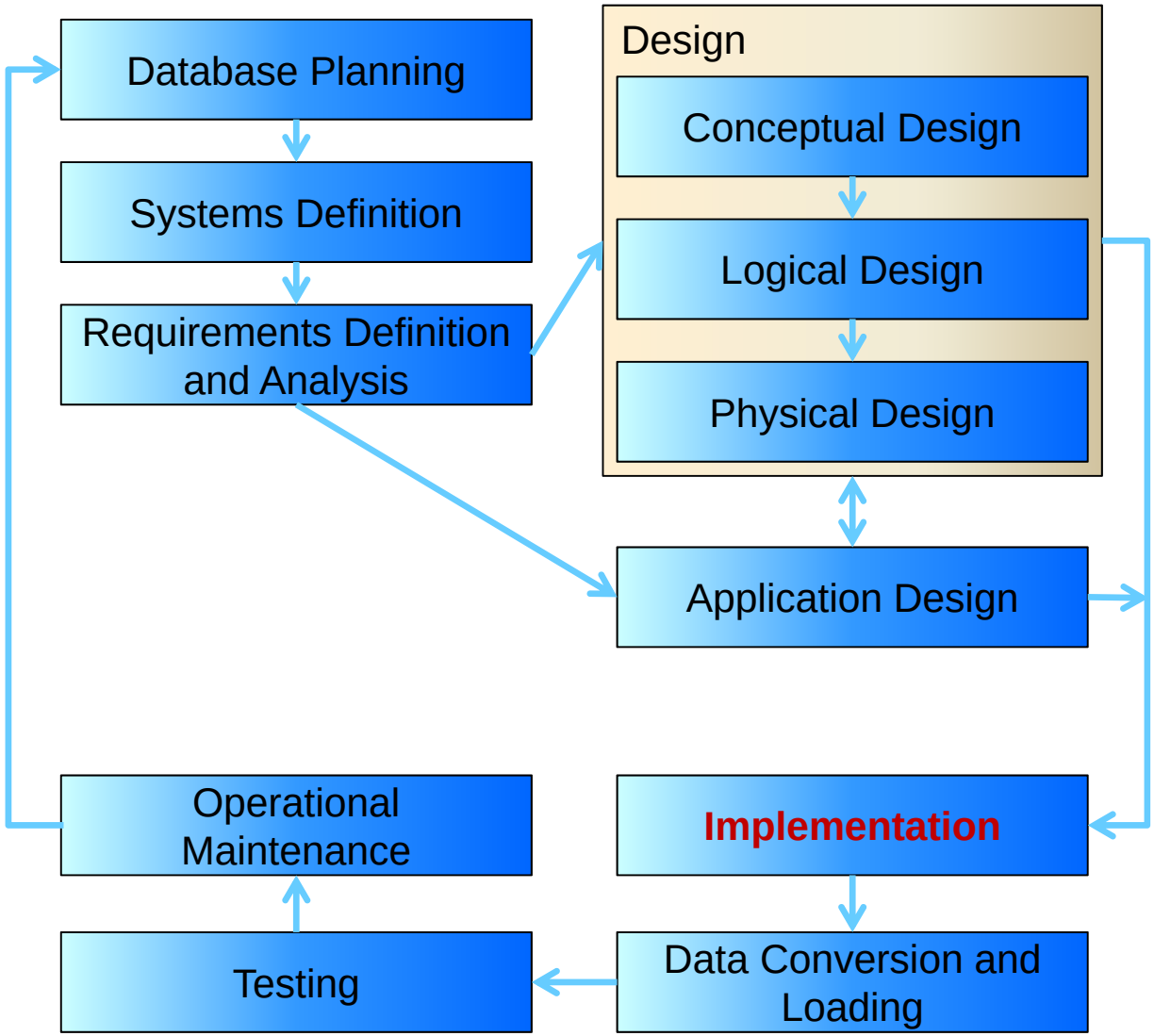
- Implements the logical design for a specific DBMS.
- Describes:
  - Base tables
  - Data types
  - Indexes
  - Integrity constraints
  - File organisation
  - Security measures
- We will cover some aspects of physical design

Database Development Lifecycle



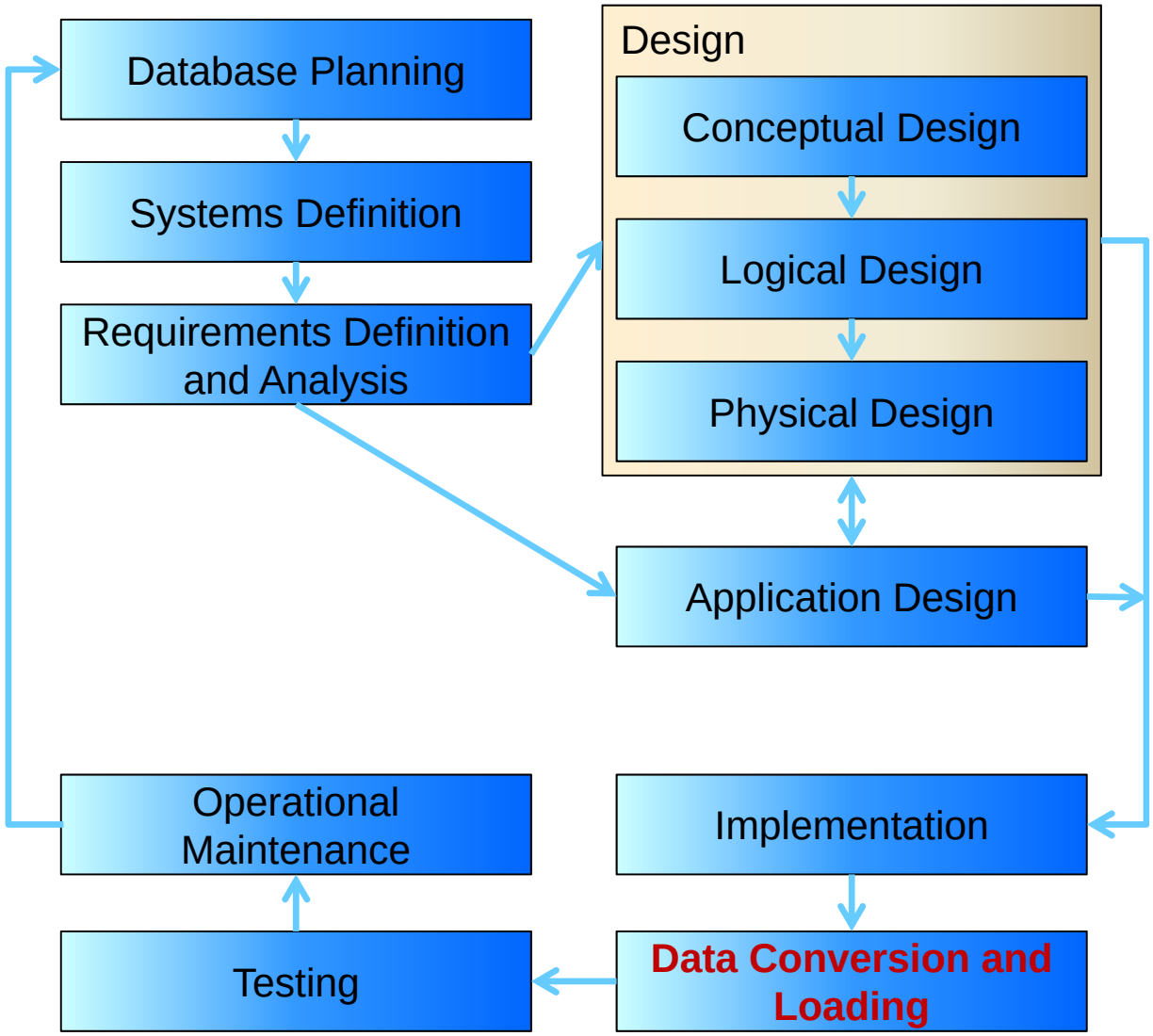
- Done in conjunction with design
- Design of the interface and application programs that use and process the database

Database Development Lifecycle



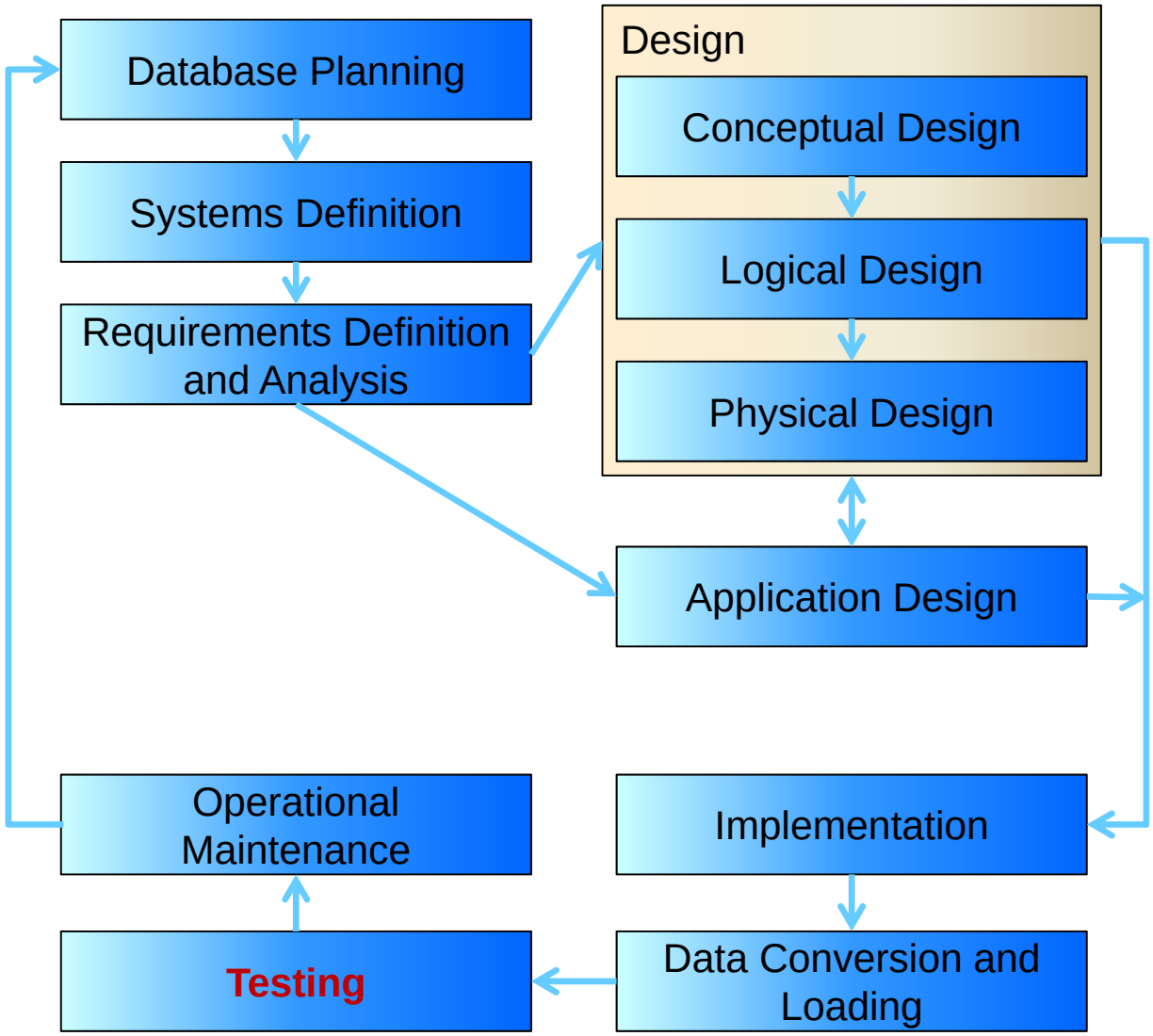
- Implementation of the design as a working database

Database Development Lifecycle



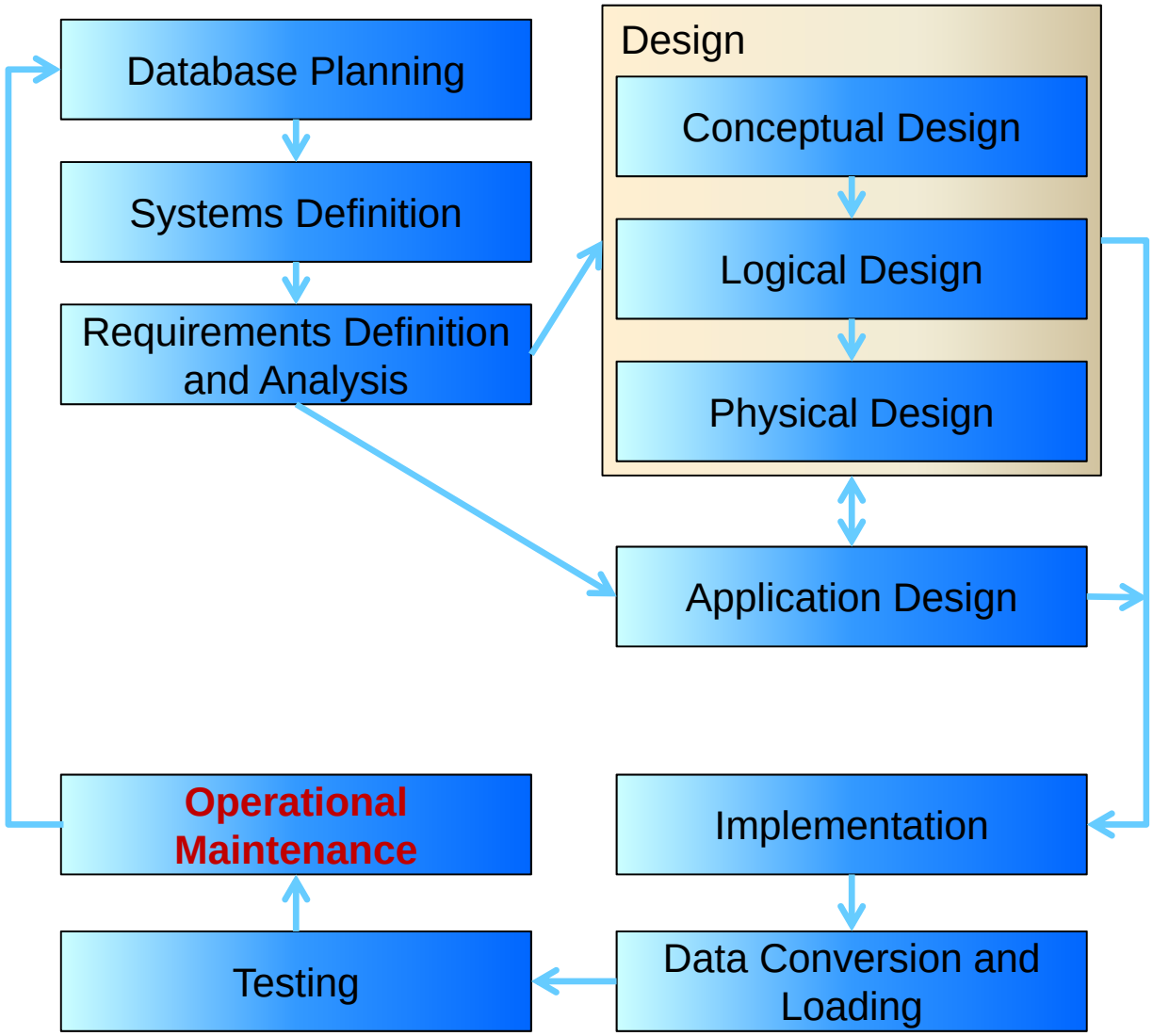
- Transfer existing data into the database
- Conversion from old systems
- Non trivial task

Database Development Lifecycle



- Running the database to find errors in the design / setup
- Other issues also
  - Performance
  - Robustness
  - Recoverability
  - Adaptability
- Outside scope of the course

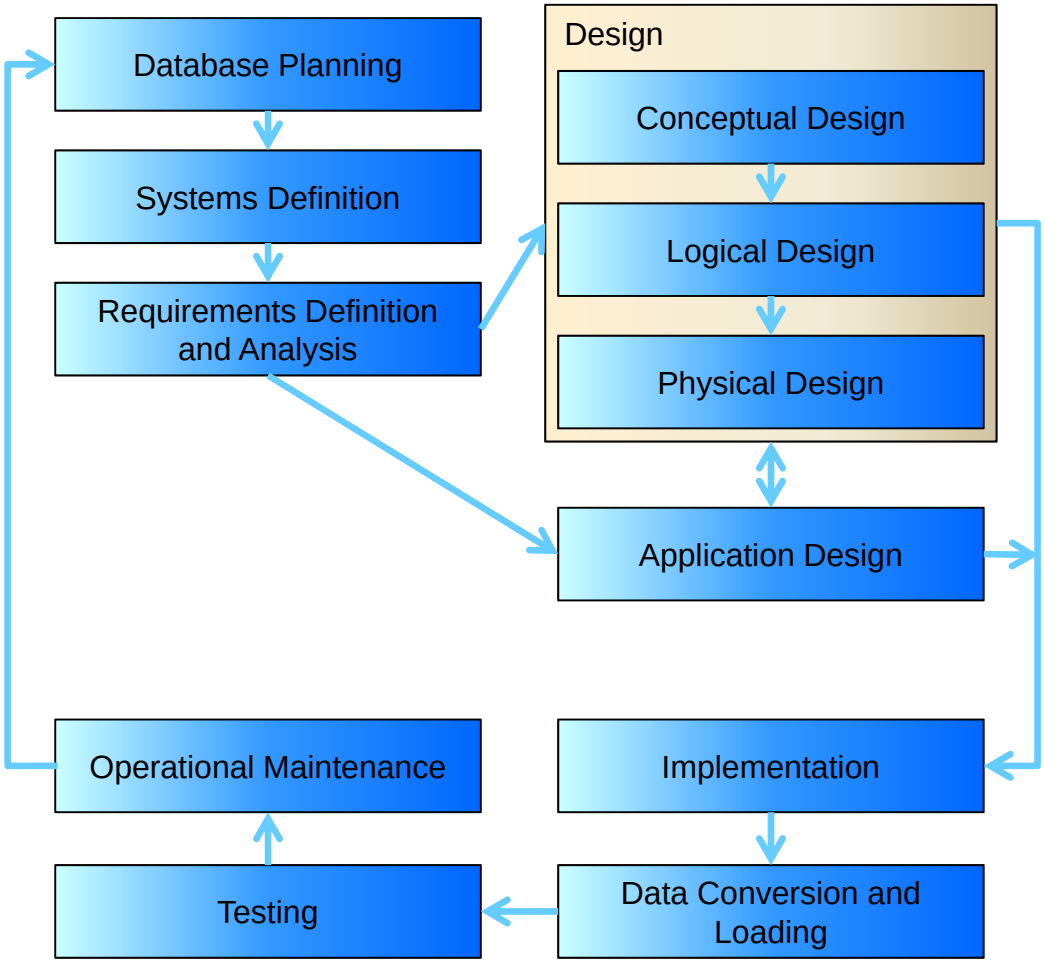
Database Development Lifecycle



- The process of monitoring and maintaining the database following its commissioning
- Monitoring and improving performance
- Handling changes to requirements

- We will touch on some of these topics later in the semester

Summary of database lifecycle



Now we'll work through one example …

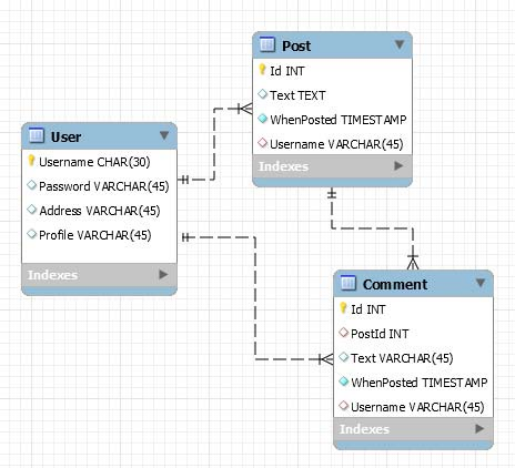
Case Study: design the db

Case for this lecture: Department store

- This database is the central component of an information system used to manage a large city department store. The store has several departments; about each we must record its name and id, phone number, and which floor it is on. Each department has several employees working for it, and one boss (an employee) who manages it.
- About each employee we record their name and id, their salary, which department they work for and which other employee is their boss.
- The items that the store sells each have a name and id, a type and a colour. Whenever a department sells an item, we record which item was sold, how many were sold, which department sold it – and we give each sale an integer id.
- Items are delivered to the store by suppliers, about each of whom we record a name and id, and a phone number. When a delivery is made, we record how many of which item was delivered by which supplier to which department – and deliveries get an integer id too.

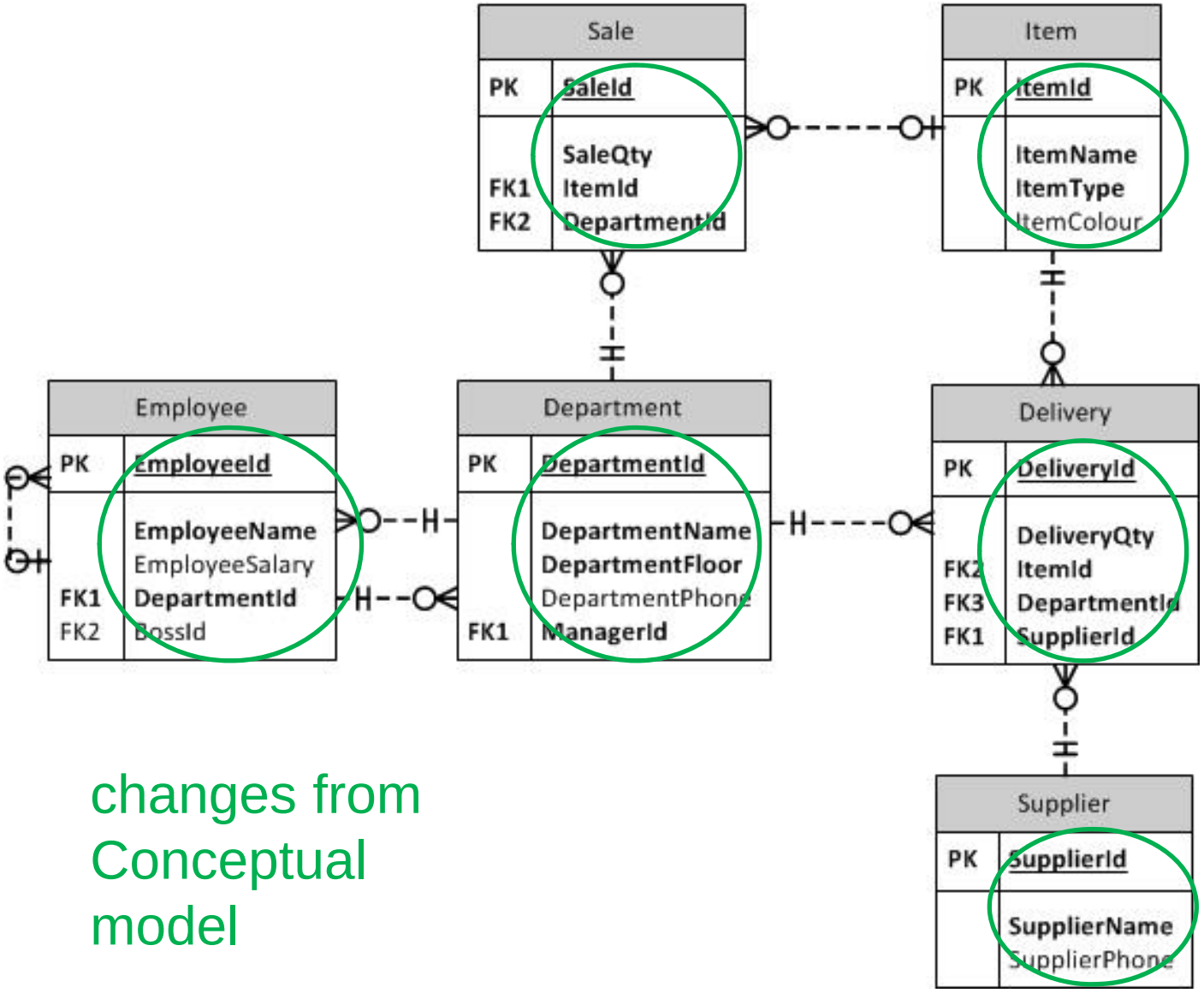
Recall from week 1: Data Modelling

1. What are the entities that need to be tracked?
2. What information will be recorded about each entity?
3. What are the relationships between entities?
4. What are the cardinalities of relationships?



Conceptual Data Model

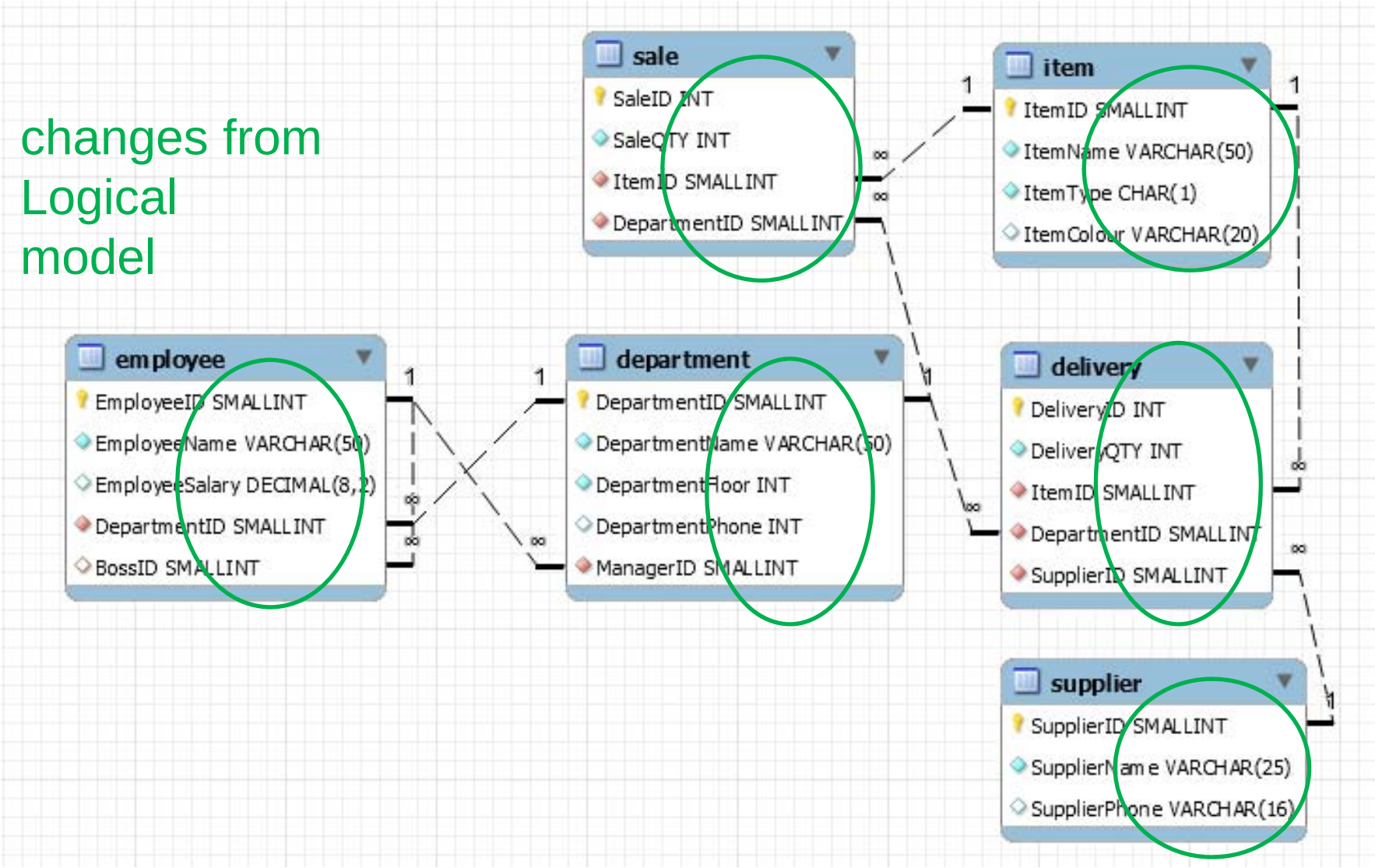
Logical Data Model



changes from Conceptual model

Physical Model

changes from Logical model



Data Dictionary

entity “Employee”

Key	Attribute	Data type	Not null	Unique	Description
PK	EmployeeId	SmallInt	Yes	Yes	Automatically assigned
	EmployeeName	Varchar(50)	Yes	No	Must have a name
	EmployeeSalary	Decimal(8,2)	No	No	Salary might be assigned later
FK	DepartmentId	SmallInt	Yes	No	Must be same as Department pk
FK	BossId	SmallInt	Yes	No	Must be same as Employee pk

Case Study: implement the db

Create tables

Can be done in several ways:

1. manual DDL commands
2. SQL "setup script" (right)
3. forward engineer from MySQL Workbench and similar tools

```
15
16  CREATE TABLE Item (
17      ItemID  SMALLINT,
18      ItemName    VARCHAR(50) NOT NULL,
19      ItemType    CHAR(1),
20      ItemColour  VARCHAR(20),
21      PRIMARY KEY (ItemID)
22  );
23
24  CREATE TABLE Employee (
25      EmployeeID      SMALLINT,
26      EmployeeName    VARCHAR(50),
27      EmployeeSalary  DECIMAL(8,2),
28      DepartmentID    SMALLINT,
29      BossID  SMALLINT,
30      PRIMARY KEY (EmployeeID),
31    FOREIGN KEY (BossID) references Employee(EmployeeID),
32    FOREIGN KEY (DepartmentID) references Department(DepartmentID)
33  );
34
35  CREATE TABLE Department (
36  DepartmentID        SMALLINT,
37  DepartmentName    VARCHAR(50)   NOT NULL,
38  DepartmentFloor     INTEGER,
39  DepartmentPhone     INTEGER,
40  ManagerID         SMALLINT   NOT NULL,
41  PRIMARY KEY (DepartmentID),
42  FOREIGN KEY (ManagerID) references Employee(EmployeeID)
43  );
44
45
```

Insert data

Can be done in several ways:

1. manual Insert commands
2. SQL script file (right)
3. Insert in WB model
4. while in Production mode, via application software

Result Grid | Filter Rows: | Edit: | Export/Import:

EmployeeID	EmployeeName	EmployeeSalary	DepartmentID	BossID
1	Alice	75000.00	1	0
2	Ned	45000.00	11	1
3	Andrew	25000.00	11	2
4	Clare	22000.00	11	2
5	Todd	38000.00	2	1
6	Nancy	22000.00	2	5
7	Brier	43000.00	9	1
8	Sarah	56000.00	9	7
9	Sophie	35000.00	10	1

employee 1

```
INSERT INTO Department VALUES
    (1,'Management',5,34,1),
    (8,'Books',1,81,4),
    (3,'Clothes',2,24,4),
    (4,'Equipment',3,57,3),
    (5,'Furniture',4,14,3),
    (6,'Navigation',1,41,3),
    (7,'Recreation',2,29,4),
    (2,'Accounting',5,35,5),
    (9,'Purchasing',5,36,7),
    (10,'Personnel',5,37,9),
    (11,'Marketing',5,38,2);


INSERT INTO Employee VALUES
    (1,'Alice',75000,1,0),
    (2,'Ned',45000,11,1),
    (3,'Andrew',25000,11,2),
    (4,'Clare',22000,11,2),
    (5,'Todd',38000,2,1),
    (6,'Nancy',22000,2,5),
    (7,'Brier',43000,9,1),
    (8,'Sarah',56000,9,7),
    (9,'Sophie',35000,10,1),
    (10,'Sanjay',15000,6,3),
    (11,'Rita',15000,8,4),
    (12,'Gigi',16000,3,4),
    (13,'Maggie',16000,3,4),
    (14,'Paul',11000,4,3),
    (15,'James',15000,4,3),
    (16,'Pat',15000,5,3),
```

Query the data

SQL Select statement

```
1  select * from employee;   /* table dump */
2  select employeename from employee; /* only one column */
3  /* only some rows */
4  select * from employee where employeesalary > 50000;
5  /* join Employee and Department tables */
6  select * from employee natural join department order by employeeid;
7  /* depts on second floor */
8  select * from department where departmentfloor = 2;
9  /* sales made by departments on second floor */
10 select * from sale natural join department
11 where departmentfloor = 2;
12 /* items sold by departments on second floor */
13 select distinct (itemid) from sale natural join department
14 where departmentfloor = 2;
15 /* show each employee's boss */
16 select emp.employeename as Employee, boss.employeename as Boss
17 from employee emp inner join employee boss
18 on emp.bossid = boss.employeeid;
```

Change data

SQL Update, Insert, Delete statements

```
1  /* total salary for employees of department 11 */
2  select sum(EmployeeSalary) from Employee where departmentid = 11;
3  /* award everyone in department 11 a 15% pay rise */
4  update Employee
5  set EmployeeSalary = EmployeeSalary * 1.15
6  where departmentid = 11;
7  /* add two new Items  */
8  select * from item;
9  insert into Item values
10 (21, 'iPhone 6', 'P', 'White'), (22, 'Nexus 6P', 'P', 'Black');
11 /* check insert */
12 select * from Item where ItemType = 'P';
13 /* delete items of type P */
14 delete from Item where ItemType = 'P';
```

Next 3 weeks

- More detailed understanding of database design
  - Conceptual design
  - Logical design
  - Physical design
- More detailed understanding of SQL
  - Single table
  - Multiple tables
  - Super- and sub-types

Homework: design a database

- The PhoneCo company provides telephony services and wants us to design a database for the Customer Billing Function of their business. The system must be able to handle billing (producing bills for customers) and payment of bills. It must capture information about customers, phone records, accounts, payments and billing details.
- A customer sets up at least one account with PhoneCo, and once approved, is given one or more phone numbers which they nominally own. The customer then uses these phone numbers, and usage charges are applied as detail lines against their account. Customers are charged a monthly line fee, a monthly equipment rental fee, and for individual phone calls. Each of these charges is governed by different rates for different customers, depending on what the customer negotiated.
- Once a month an invoice is generated and sent to the customer. The customer then pays the invoice (not necessarily in full) using a credit or debit card. Each payment made is applied against their account.

Assignment 1

- released tonight
- groups of 2 or 3
- due in week 7