



# INFO 90002

## Database Systems & Information Modelling

Week 01

Designing and Implementing a Database (example)



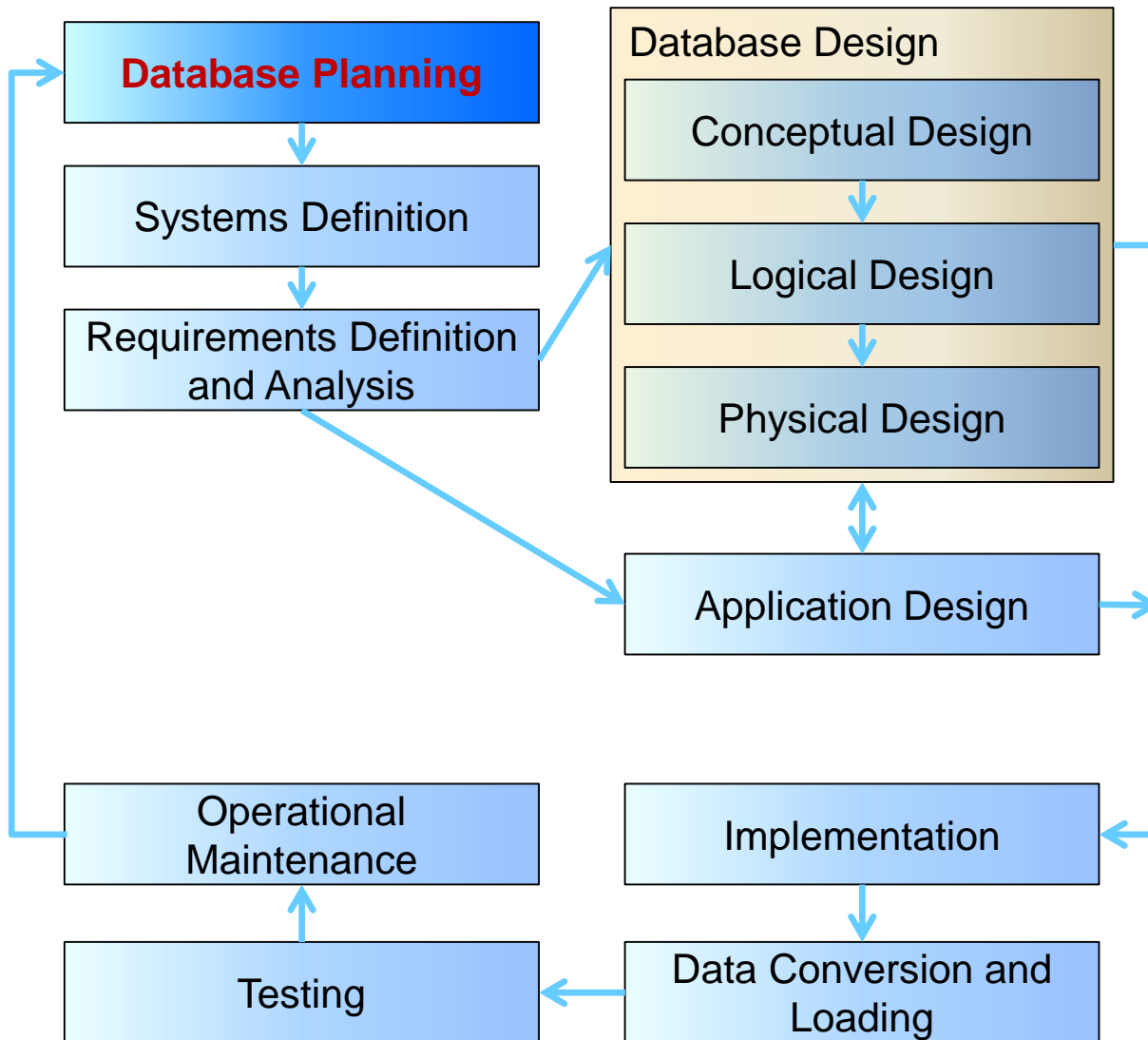
- first half: **Designing Databases**
  - the database life-cycle
  - modelling a database for an example business
    - conceptual model
    - logical model
    - physical model
- second half: **Implementing and Using Databases**
  - create the database and tables
  - populate tables with data
  - query data
  - change data



THE UNIVERSITY OF  
MELBOURNE

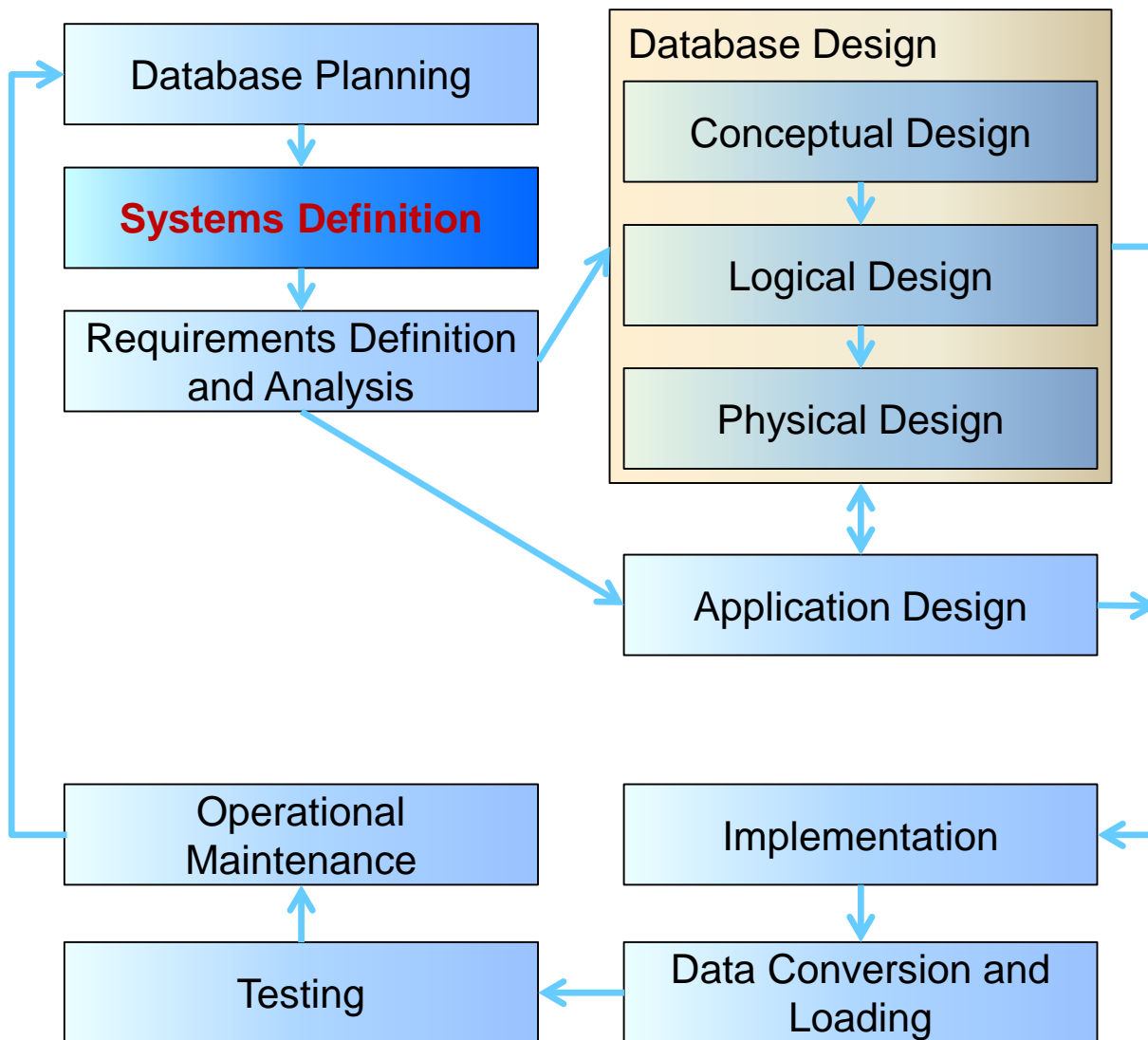
# Database Development Lifecycle

# 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

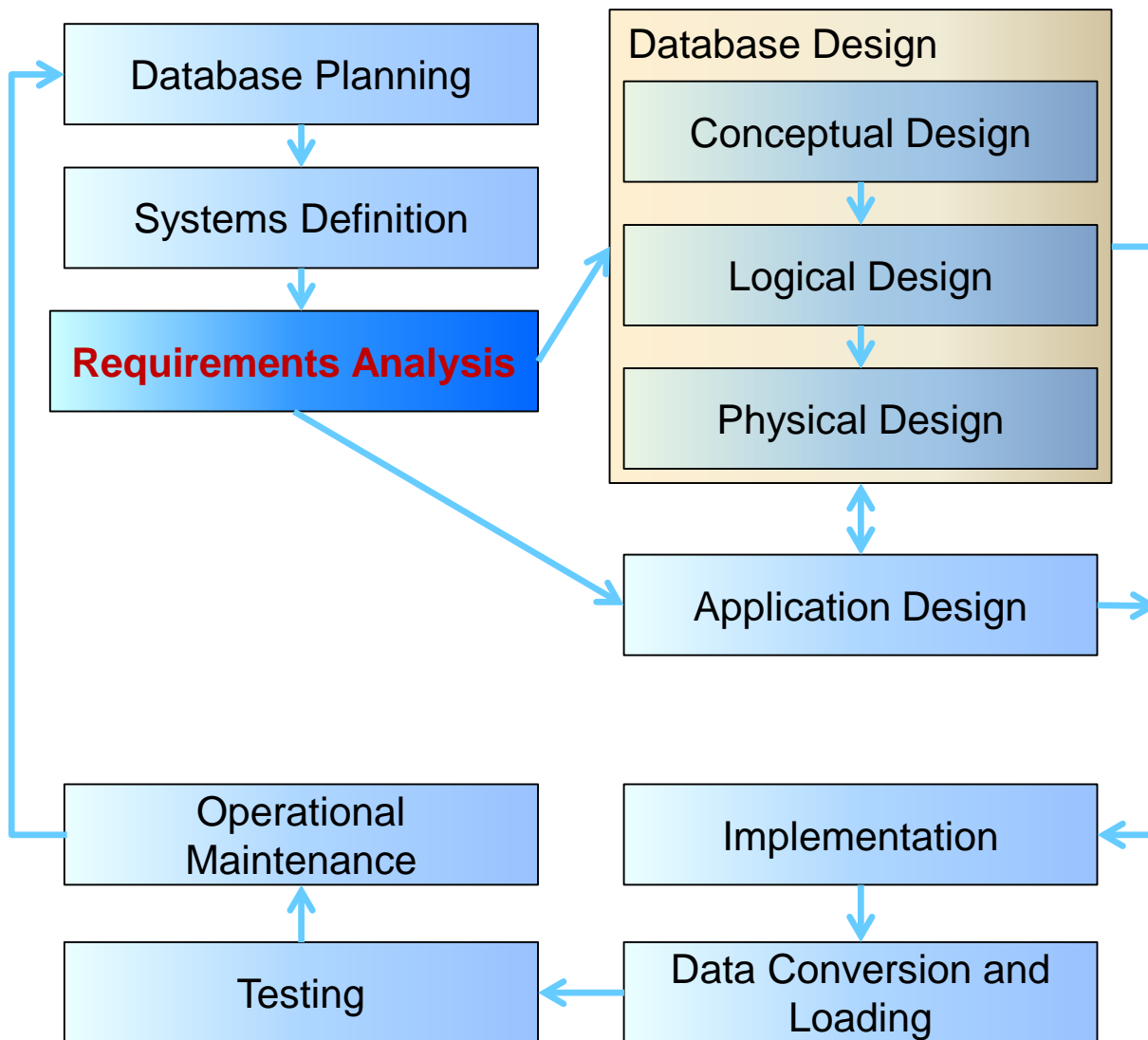
# Database Development Lifecycle



- 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)

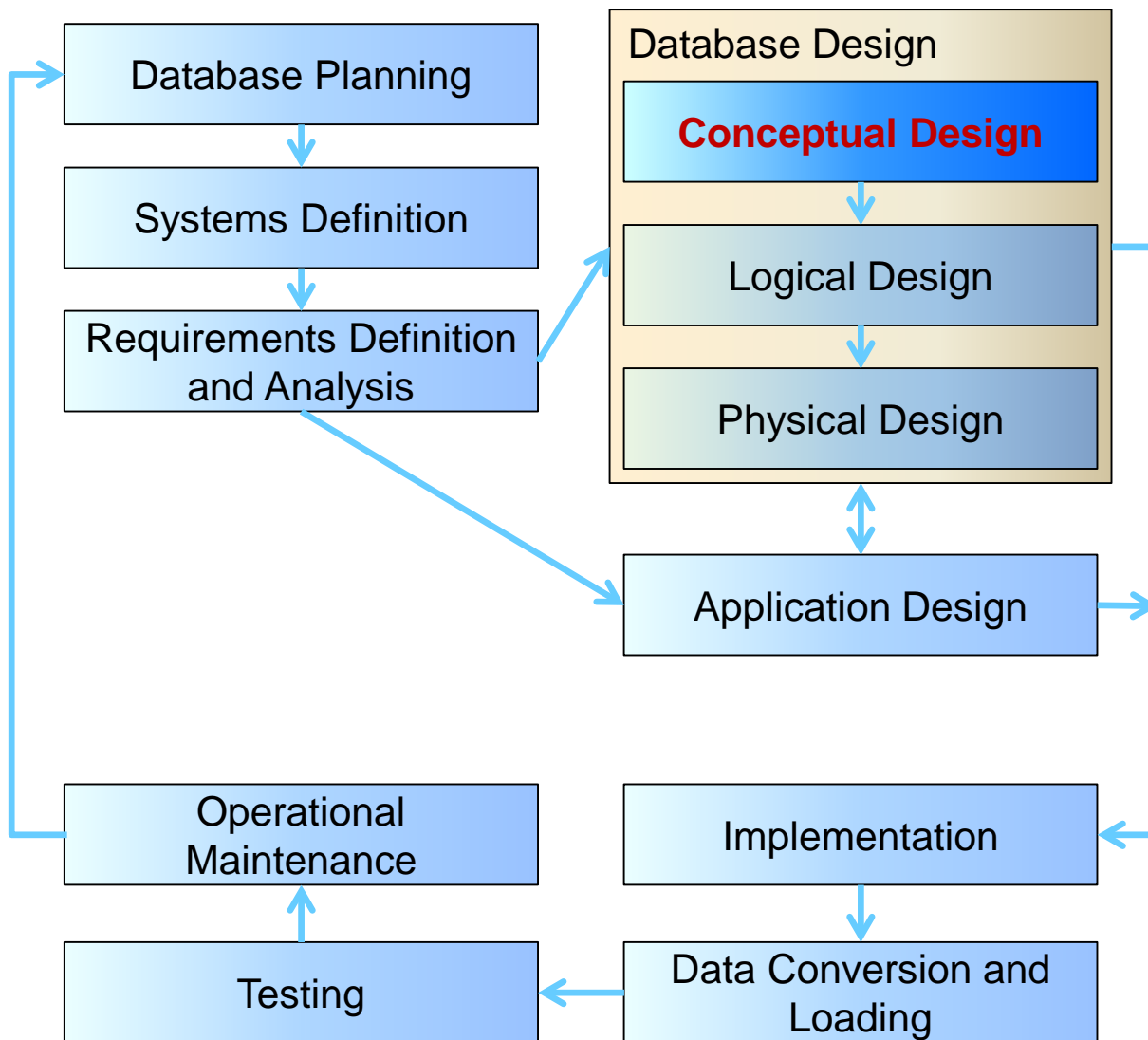


# Database Development Lifecycle

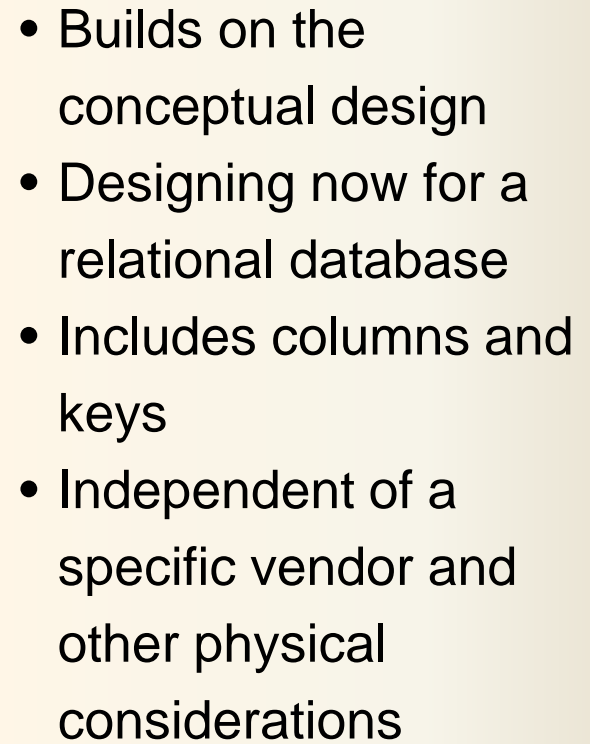


- Collection and analysis of requirements for the new system

# Database Development Lifecycle

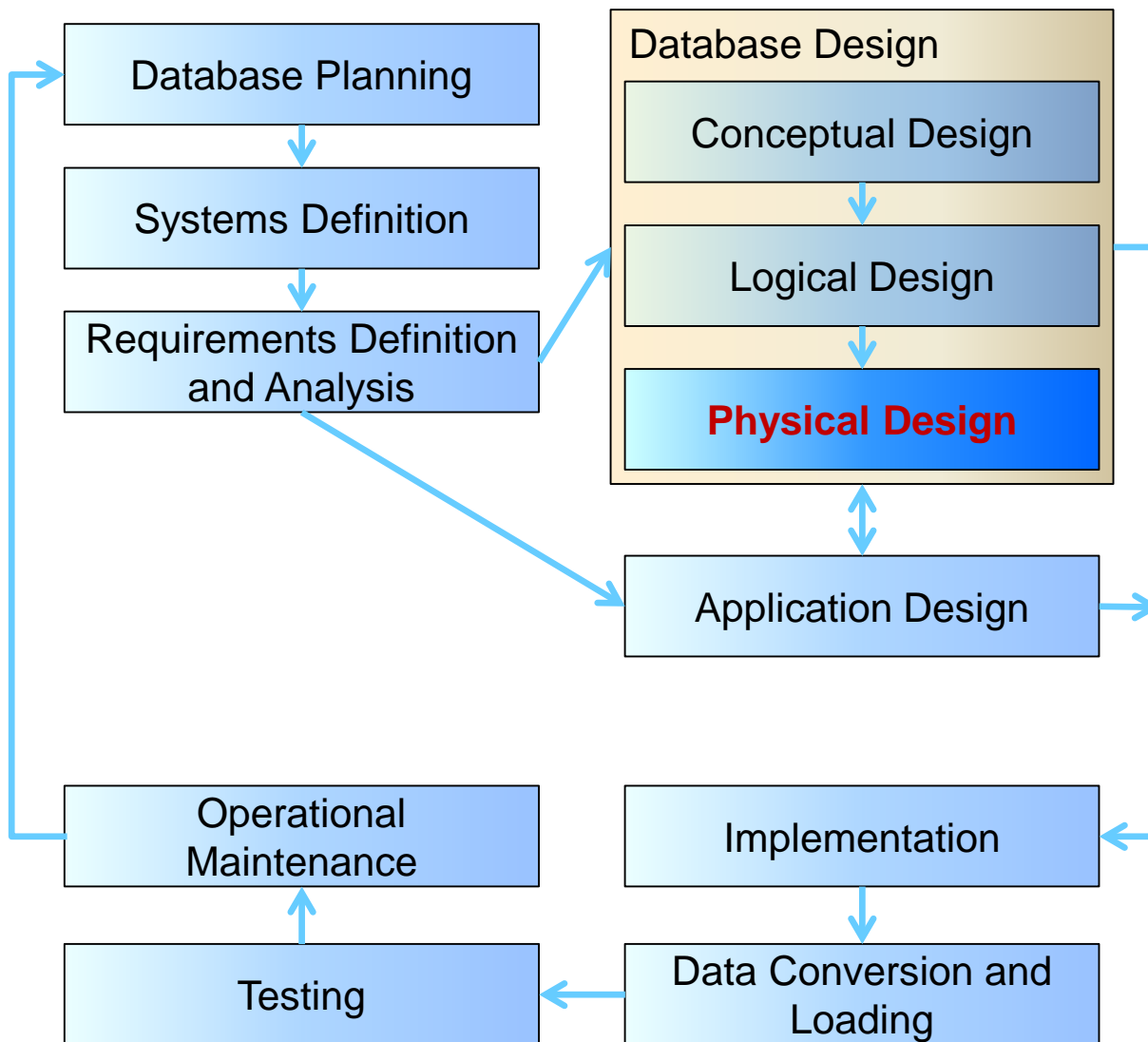


- High-level, first-pass model of entities and their connections
- Typically omits attributes
- Could potentially be implemented in a non-relational database
- Thus can include many-to-many relationships, repeating groups, composite attributes



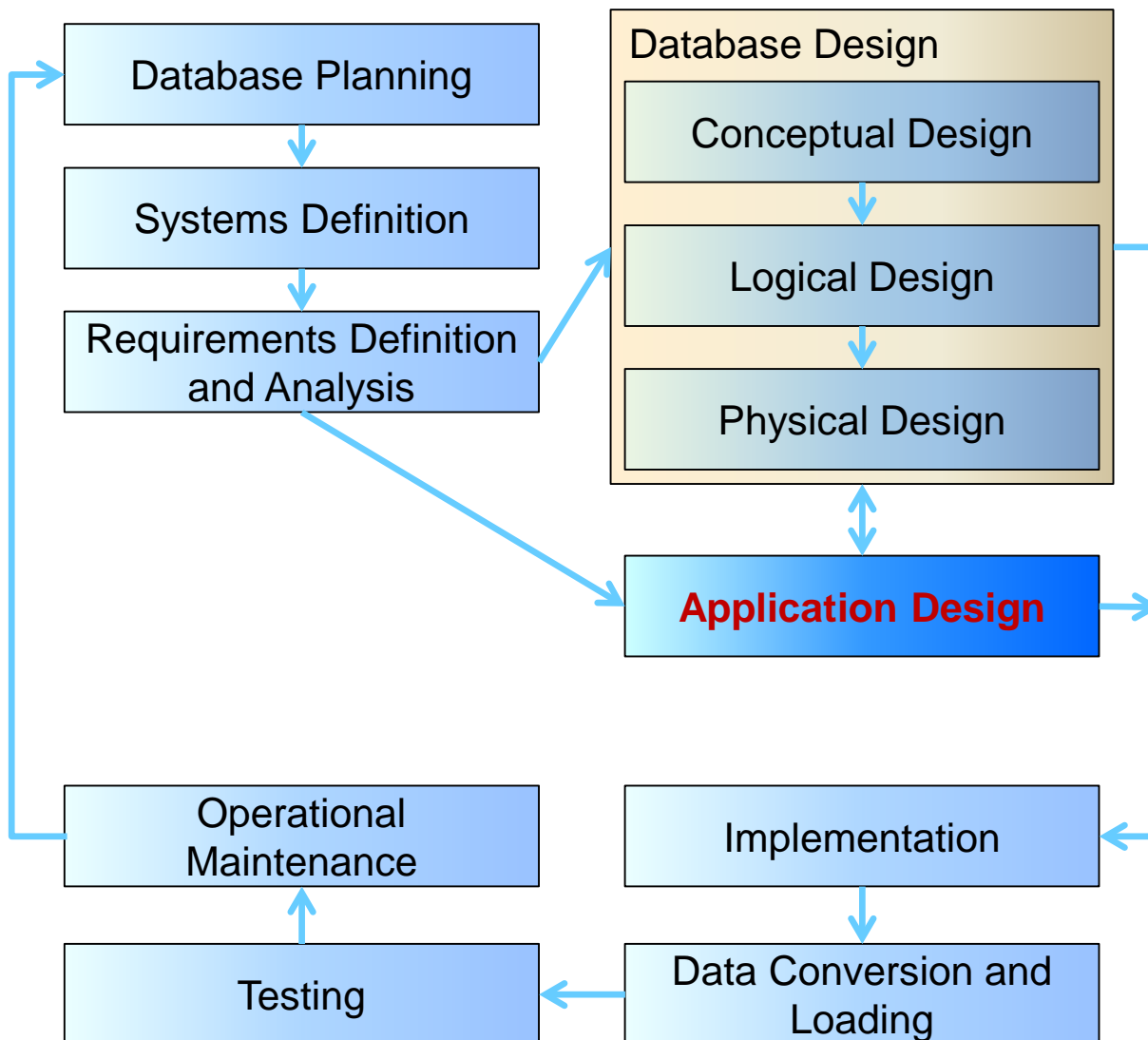


# Database Development Lifecycle



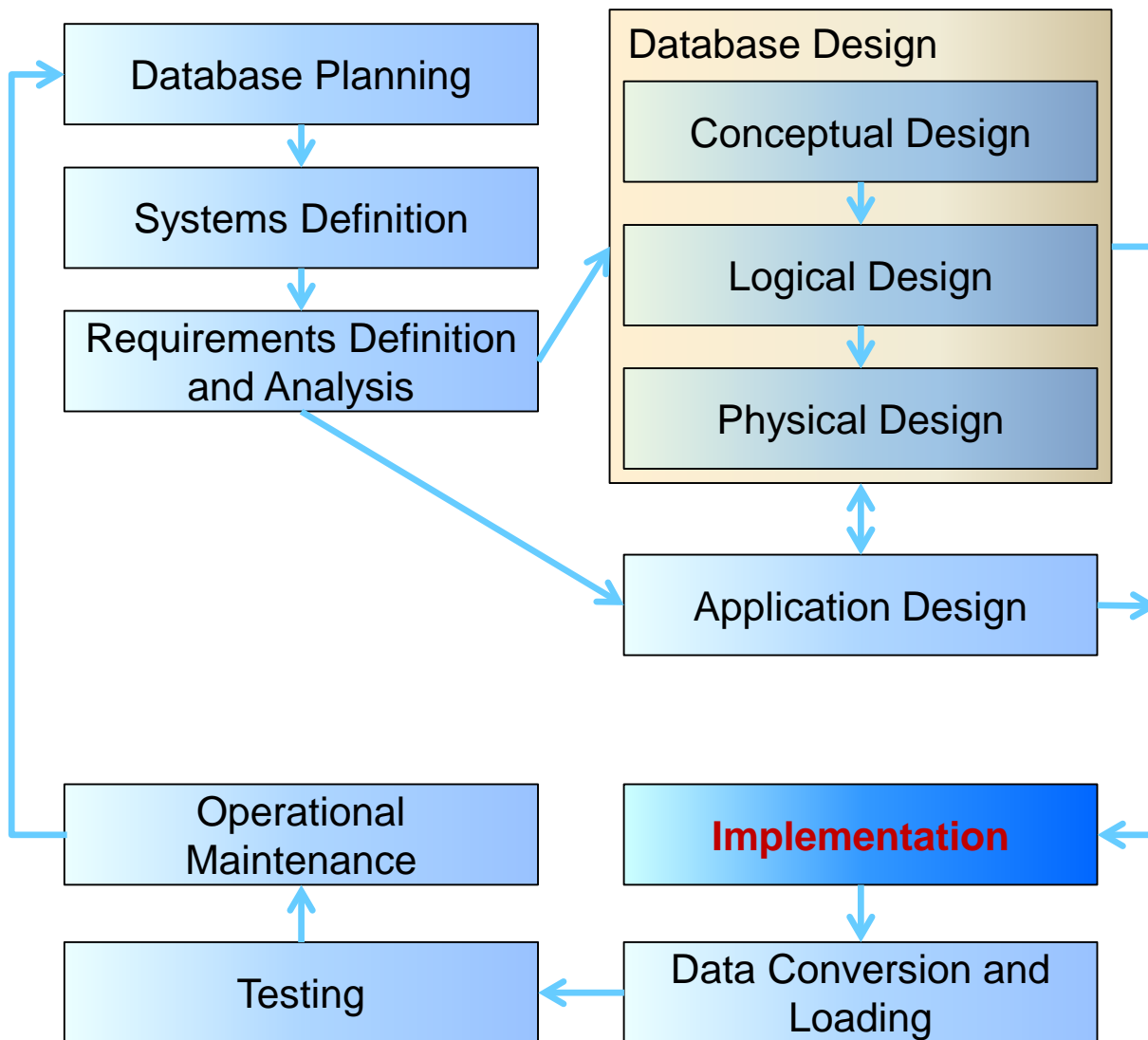
- Implements the logical design for a specific DBMS.
- Describes:
  - Base tables
  - Data types
  - Indexes
  - Integrity constraints
  - File organisation
  - Security measures
- We will cover some aspects of physical design

# Database Development Lifecycle



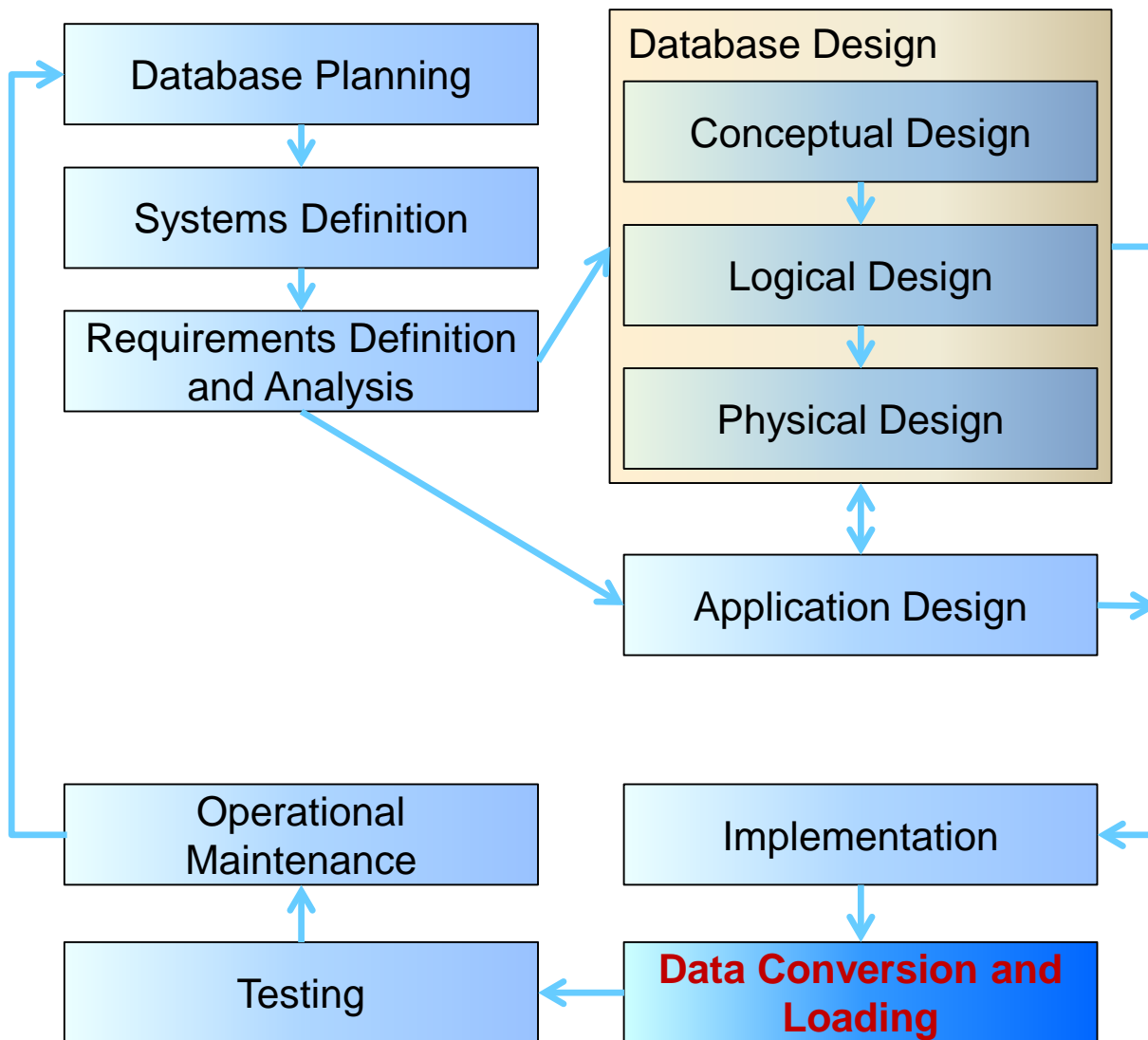
- Done in conjunction with database design
- Design of the interface and application programs that use and process the database
- Mostly outside scope of the course, but discussed in week 7

# Database Development Lifecycle



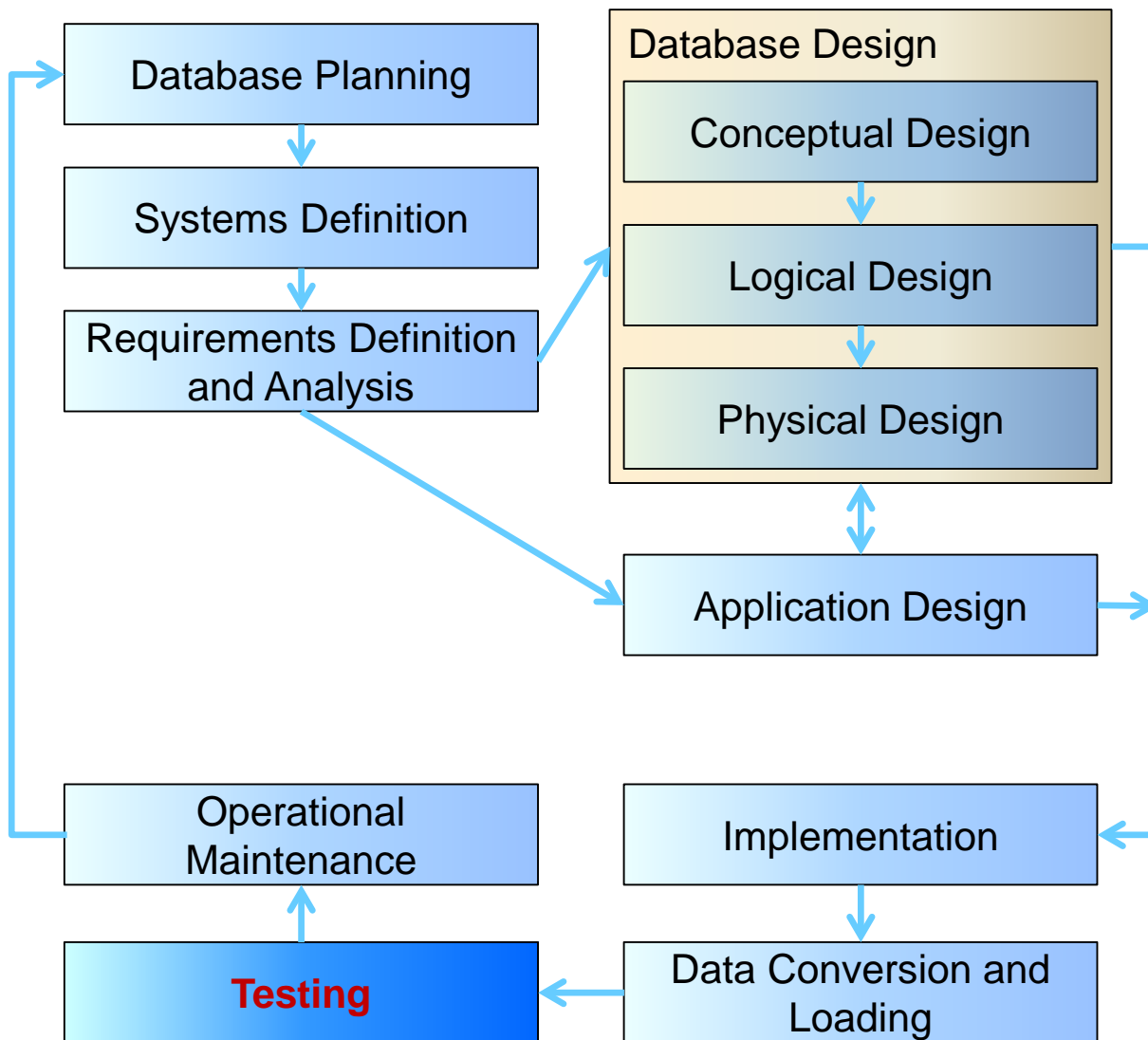
- Implementation of the design as a working database

# Database Development Lifecycle

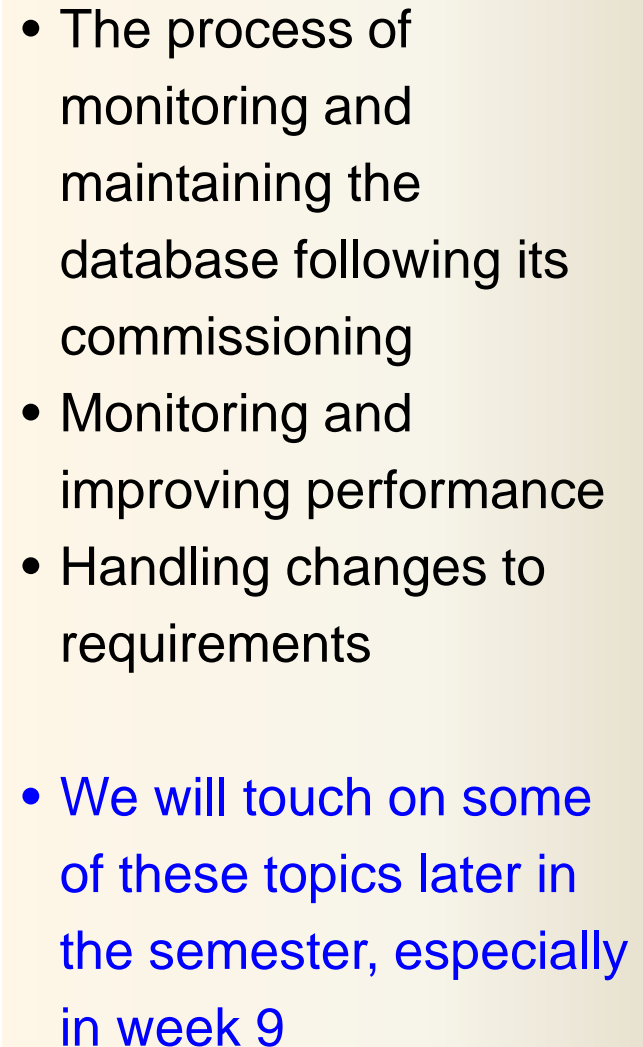


- Transfer existing data into the database
- Conversion from old systems
- Non trivial task

# Database Development Lifecycle

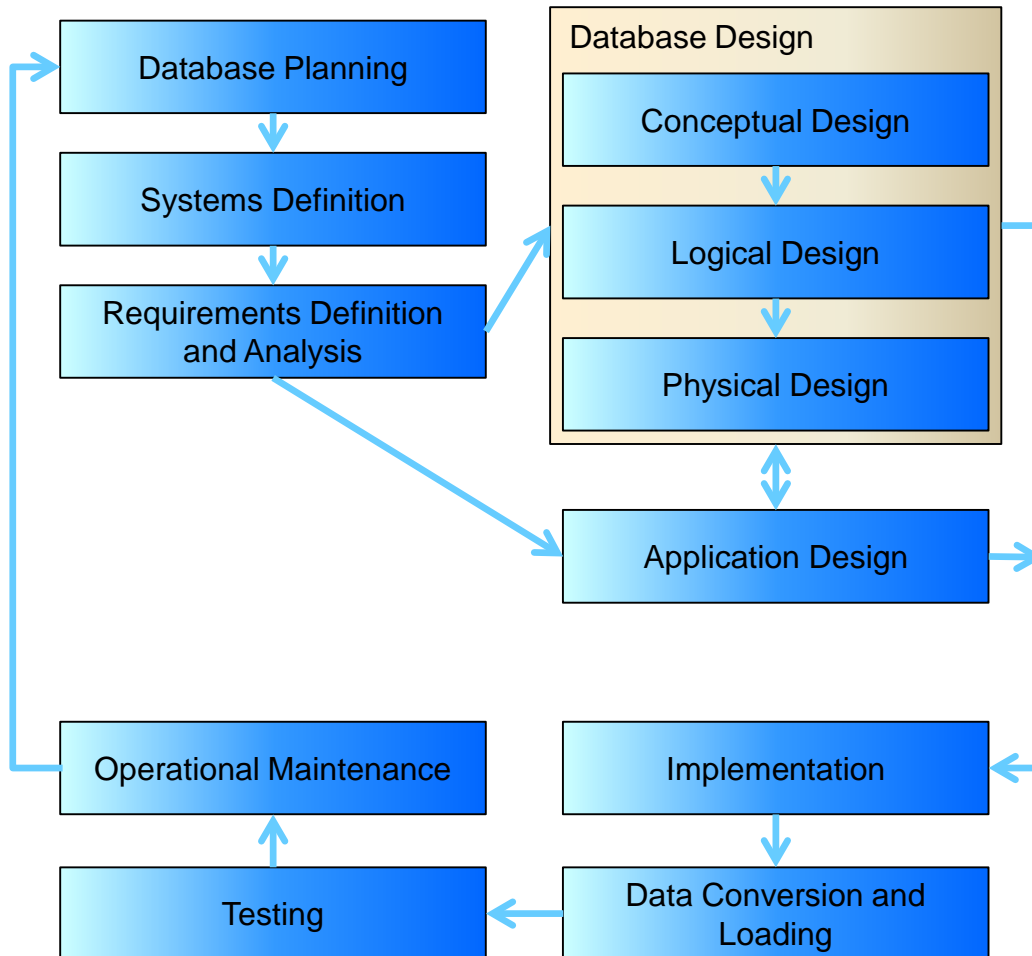


- Running the database to find errors in the design / setup
- Other issues also
  - Performance
  - Robustness
  - Recoverability
  - Adaptability
- Mostly outside scope of the course





# Summary of database lifecycle



Now we'll work through one example ...



# Case Study: design the db

## Data Modelling



# Case for this lecture: Orders system

- Our company sells many products. About each product we record its id, name, and price.
- We have many customers. About each customer we record their customer id, name, and address.
- Customers place orders for products. Each order is placed by one customer on a particular date. Over time a customer may place several orders, though some may register but not place any orders.
- Each order must contain at least one order-item, but may contain several. Each order-item records a quantity ordered of one product.

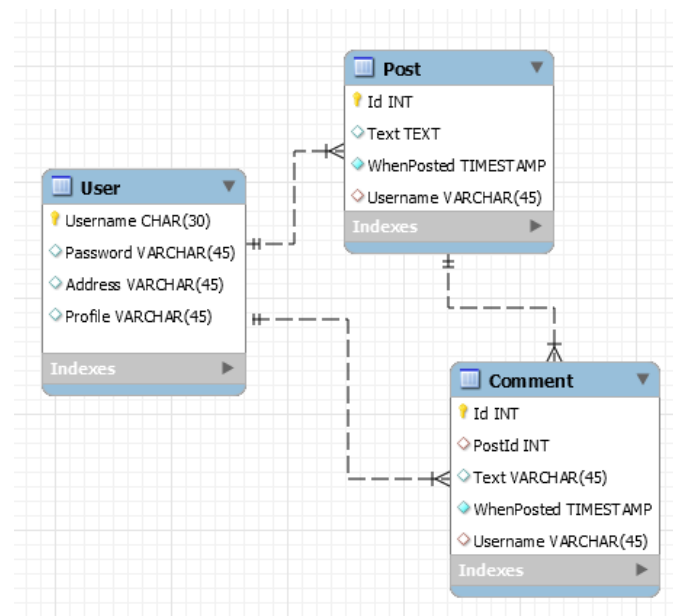
**Order Form**
3-Oct

Ship to:

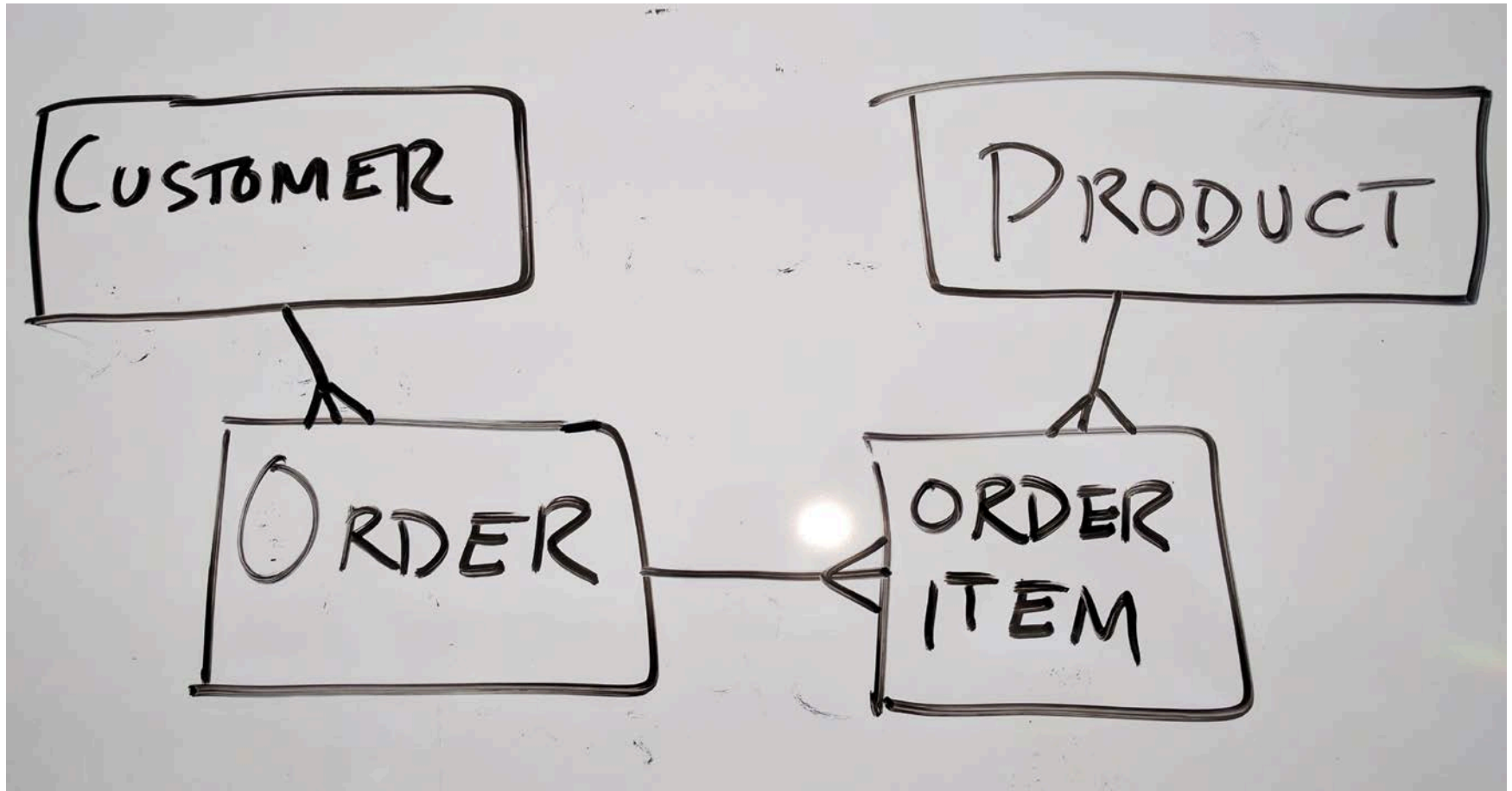
June Summers  
 123 Main St  
 Toronto, ON M5M 5M5

Product	Price	Qty	Total
Sweater	\$ 15.00	5	\$ 75.00
<div> <div></div> <div>           Jacket            Sweater  <b>Shirt</b>            Pants            Dress         </div> </div>			
Total			\$ 75.00

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?



(we will create this together during the lecture)

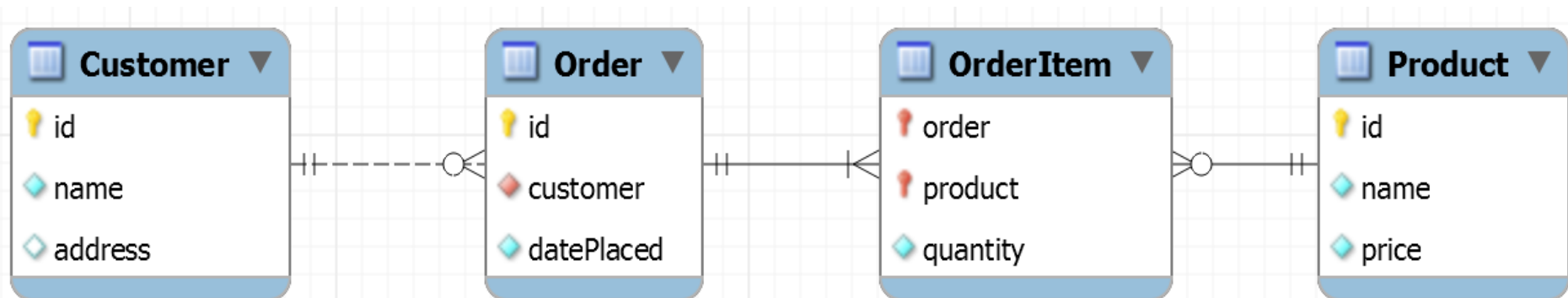




(we will create this together during the lecture)

New question:

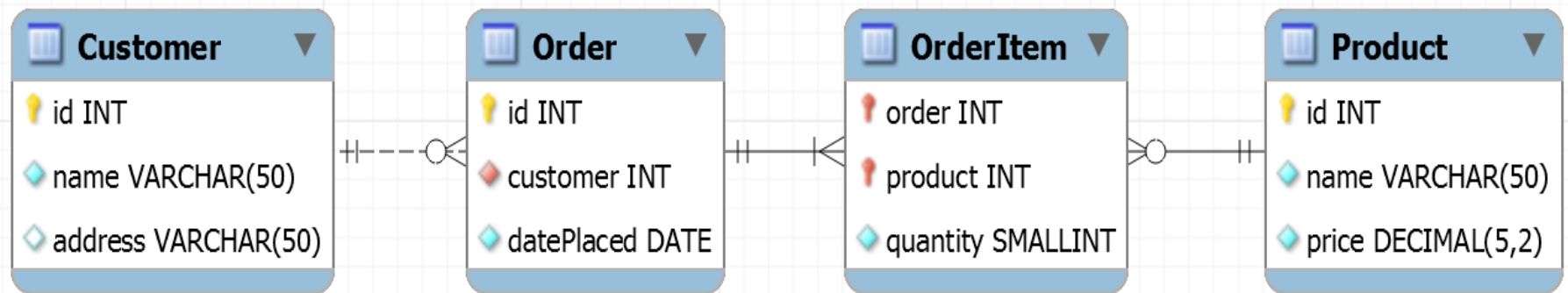
What is the *primary key* of each table?



(we will create this together during the lecture)

New question:

What is the *data type* of each column?





# Case Study: implement the db

## SQL

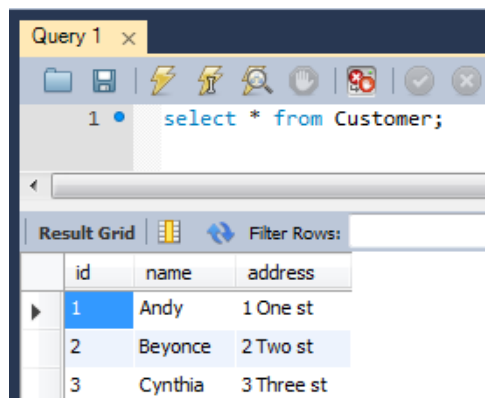
Can be done in several ways:

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

```
10
11 CREATE TABLE IF NOT EXISTS `Customer` (
12   `id` INT NOT NULL,
13   `name` VARCHAR(50) NOT NULL,
14   `address` VARCHAR(50) NULL,
15   PRIMARY KEY (`id`))
16 ENGINE = InnoDB;
17
18
19 CREATE TABLE IF NOT EXISTS `Order` (
20   `id` INT NOT NULL,
21   `customer` INT NOT NULL,
22   `datePlaced` DATE NOT NULL,
23   PRIMARY KEY (`id`),
24   INDEX `fk_Order_Customer_idx` (`customer` ASC),
25   CONSTRAINT `fk_Order_Customer`
26     FOREIGN KEY (`customer`)
27       REFERENCES `Customer` (`id`)
28     ON DELETE NO ACTION
29     ON UPDATE NO ACTION)
30 ENGINE = InnoDB;
31
32
33 CREATE TABLE IF NOT EXISTS `Product` (
34   `id` INT NOT NULL,
35   `name` VARCHAR(50) NOT NULL,
36   `price` DECIMAL(5,2) NOT NULL,
37   PRIMARY KEY (`id`))
38 ENGINE = InnoDB;
39
40
41 CREATE TABLE IF NOT EXISTS `OrderItem` (
42   `order` INT NOT NULL,
43   `product` INT NOT NULL,
44   `quantity` SMALLINT NOT NULL,
45   PRIMARY KEY (`order`, `product`),
46   INDEX `fk_OrderItem_Product1_idx` (`product` ASC),
47   CONSTRAINT `fk_OrderItem_Order1`
48     FOREIGN KEY (`order`)
49       REFERENCES `Order` (`id`)
50     ON DELETE NO ACTION
51     ON UPDATE NO ACTION,
52   CONSTRAINT `fk_OrderItem_Product1`
53     FOREIGN KEY (`product`)
54       REFERENCES `Product` (`id`)
55     ON DELETE NO ACTION
56     ON UPDATE NO ACTION)
57 ENGINE = InnoDB;
58
```

Can be done in several ways:

1. manual Insert commands
2. SQL script file (right)
3. Insert in Workbench model
4. via application software



Query 1 x

1 • `select * from Customer;`

Result Grid

	id	name	address
▶	1	Andy	1 One st
	2	Beyonce	2 Two st
	3	Cynthia	3 Three st

```
/* INSERT DATA */

insert into Customer values (1,'Andy','1 One st');
insert into Customer values (2,'Beyonce','2 Two st');
insert into Customer values (3,'Cynthia','3 Three st');

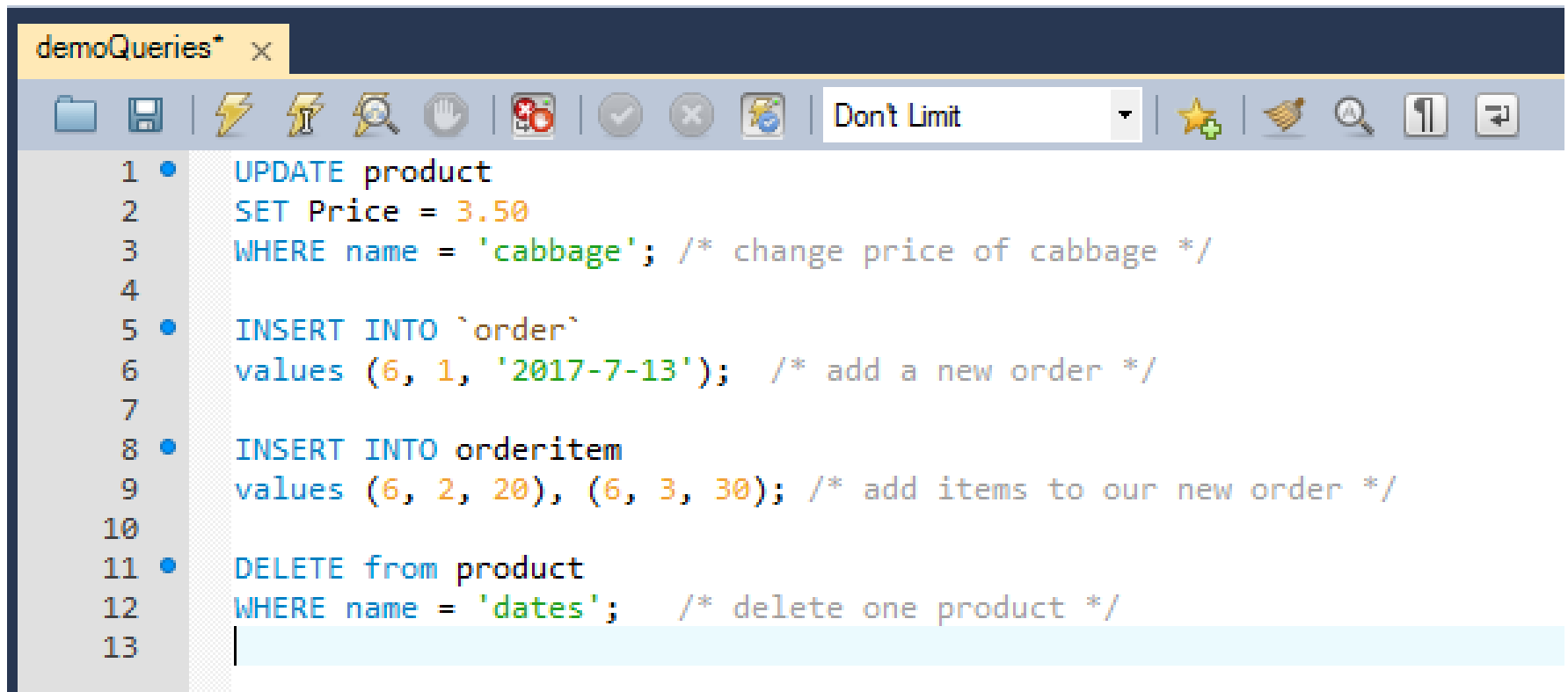
insert into Product values (1,'apple',1.11);
insert into Product values (2,'bread',2.22);
insert into Product values (3,'cabbage',3.33);
insert into Product values (4,'dates',4.44);

insert into `Order` values (1,1,'2017-6-1');
insert into `Order` values (2,2,'2017-6-2');
insert into `Order` values (3,3,'2017-6-3');
insert into `Order` values (4,1,'2017-6-4');
insert into `Order` values (5,2,'2017-6-5');

insert into OrderItem values (1,1,1);
insert into OrderItem values (1,2,2);
insert into OrderItem values (1,3,3);
insert into OrderItem values (2,2,4);
insert into OrderItem values (2,4,5);
insert into OrderItem values (3,1,6);
insert into OrderItem values (3,2,7);
insert into OrderItem values (3,3,8);
insert into OrderItem values (3,4,9);
insert into OrderItem values (4,4,10);
insert into OrderItem values (5,3,9);
```



```
IntroToMySQL x
1 • SELECT name, price FROM product
2   WHERE price > 3;
3
4 • SELECT * FROM customer
5   ORDER BY name DESC;
6
7 • SELECT AVG(price) FROM PRODUCT;
8
9 • SELECT * FROM orderitem JOIN product
10  ON orderitem.product = product.id;
11
12 • SELECT product, COUNT(*), SUM(price)
13   FROM orderitem JOIN product
14   ON orderitem.product = product.id
15   GROUP BY product;
16
```



```
demoQueries* x
1 • UPDATE product
2   SET Price = 3.50
3   WHERE name = 'cabbage'; /* change price of cabbage */
4
5 • INSERT INTO `order`
6   values (6, 1, '2017-7-13'); /* add a new order */
7
8 • INSERT INTO orderitem
9   values (6, 2, 20), (6, 3, 30); /* add items to our new order */
10
11 • DELETE from product
12   WHERE name = 'dates'; /* delete one product */
13
```



- More detailed understanding of database design
- More detailed understanding of SQL