COSC 304
Introduction to Database Systems

Database Design

Dr. Ramon Lawrence
University of British Columbia Okanagan
ramon.lawrence@ubc.ca

Database Design

The ability to design databases and associated applications is critical to the success of the modern enterprise.

Database design requires understanding both the operational and business requirements of an organization as well as the ability to model and realize those requirements in a database.

Developing database and information systems is performed using a ***development lifecycle***, which consists of a series of steps.

The Importance of Database Design

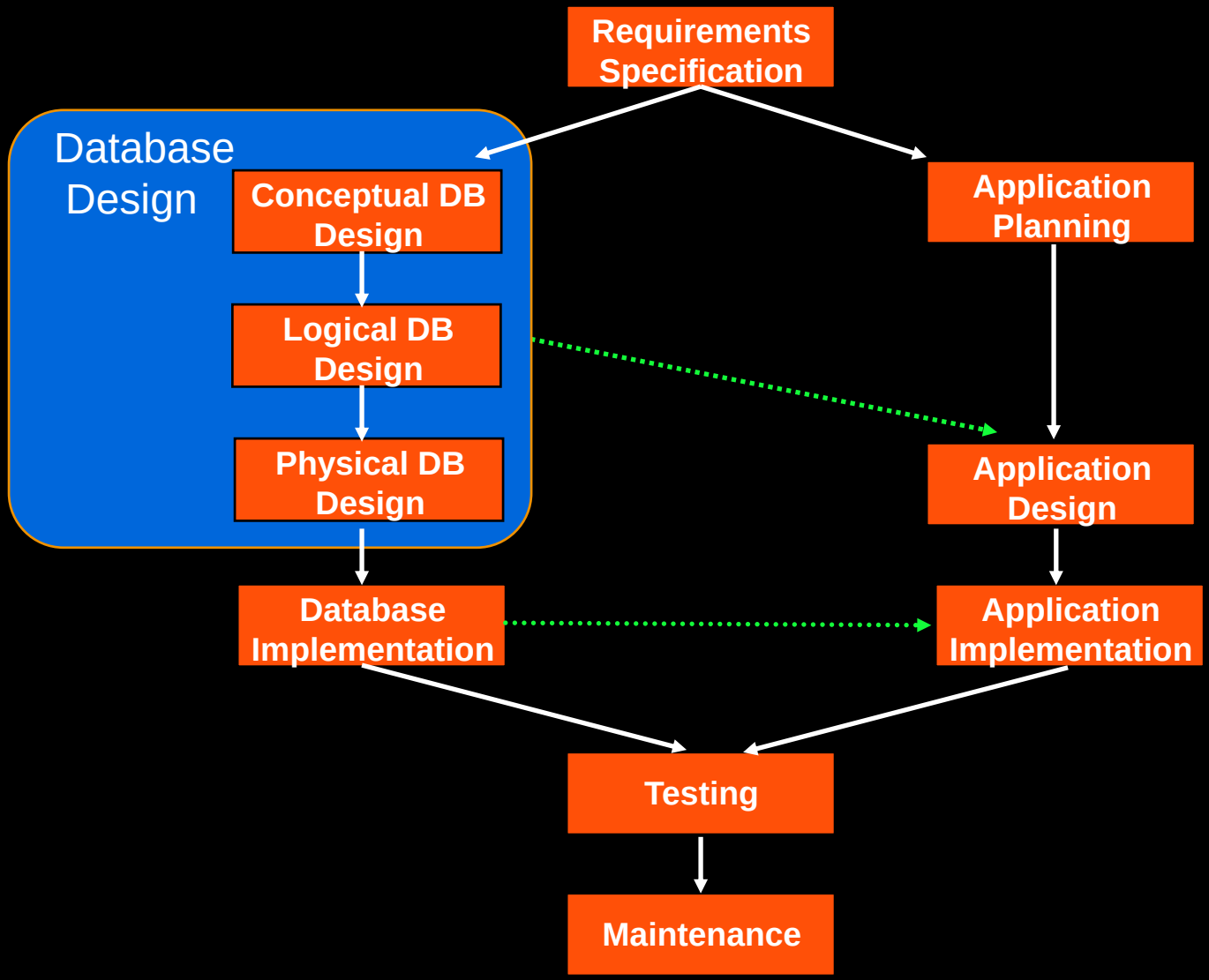
Just as proper design is critical for developing large applications, success of database projects is determined by the effectiveness of database design.

Some statistics on software projects:

- 80 - 90% do not meet their performance goals
- 80% delivered late and over budget
- 40% fail or abandoned
- 10 - 20% meet all their criteria for success
- Have you been on a project that failed? A) Yes B) No

The primary reasons for failure are improper requirements specifications, development methodologies, and design techniques.

Database Design Stages



Specification

One of the primary reasons for a project to fail is that it is not clear what the project should accomplish.

Specification involves *detailed* documentation of:

- understanding how the project fits into the organization
- project goals and success outcomes
- project timelines and deliverables
- measurement criteria for determining project success
- information on users and user requirements

Organizational planning is an early stage of specification where enterprise needs and goals are identified, new technology is evaluated, and new development projects are considered.

Specification
Mission Statements

The ***mission statement*** of a database project is a specification of the major aims and objectives of the project.

- Each mission objective should be stated clearly and preferably have defined metrics to evaluate if it is successfully completed.

The IT manager should construct the mission statement and objectives after consulting with management and users.

- The mission statement may also include:
  - the "Project Champion"
  - the estimated costs and how these costs will be paid
  - standards for database/application development that must be followed
  - documentation of privacy and security concerns
  - legal issues including copyright when dealing with outside developers
  - information on how the project will interface with other systems

A consultant is often asked to bid on a project given its mission statement and must carefully read any legal concerns.

Specification
Mission Statement Example

NASA's mission statement when going to the moon:

"I believe that this nation should commit itself to achieving the goal, before this decade is out, of landing a man on the moon and returning him safely to Earth."

- (John F. Kennedy May 25, 1961)

NASA fulfilled that goal on July 20, 1969, when Apollo 11's lunar module *Eagle* touched down in the Sea of Tranquility, with Neil Armstrong and Buzz Aldrin aboard. A dozen men would walk on the moon before the Apollo program ended. The last of those men, Gene Cernan, left the desolate lunar surface with these words: "We leave as we came and, God willing, as we shall return, with peace and hope for all mankind."

Specification
The "Project Champion"

The "**Project Champion**" is a manager or senior IT person who is the project's promoter and backer.

Many projects fail because no one takes ownership of them.

- Consequently, they take too long, go over budget, and are never deployed effectively.
- When hiring outside consultants, make sure somebody in the organization is the Project Champion.
- For internal projects, a Project Champion is especially important as there are always conflicts over money, developer time, and political issues on making users work on new applications.

Bottom line: If no one is willing to be the champion for a project, it is likely that project will not achieve its goals.

Specification Requirements Gathering

Requirements gathering collects details on organizational processes and user issues.

- Often the organization itself does not know this information, and it can only be determined by collecting it from user interviews.
- Through user interviews identify:
  - Who are the users? Group them into classes.
  - What do the users do now? (existing systems/processes)
  - What are the complaints and possibilities for improvement?
- Determine the data used by the organization, identify data relationships, and determine how data is used and generated.
  - Identify unique fields (keys)
  - Determine data dependencies, relationships, and constraints (high-level)
  - Estimate the data sizes and their growth rates

Specification Requirements Gathering (2)

The best way to determine user requirements is to:

ASK THEM, THEN LISTEN

The biggest mistake that you can make is to walk into a company and say that:

"I am going to build you a C# application using Microsoft SQL Server. Give me the data that you have, and I will go away and build you a great system!"

RULE #1: The user (and managers) are your ultimate consumers; listen and respect their opinions and needs.

RULE #1a: Users (and managers) do not always know what they want or may want the wrong things. Sometimes you must convince them of other alternatives or provide solutions that are good compromises.

Database Design

The requirements gathering and specification provides you with a high-level understanding of the organization, its data, and the processes that you must model in the database.

Database design involves constructing a suitable model of this information.

Since the design process is complicated, especially for large databases, database design is divided into three phases:

- Conceptual database design
- Logical database design
- Physical database design

Database Design
Conceptual Database Design

Conceptual database design involves modeling the collected information at a high-level of abstraction without using a particular data model or DBMS.

High-level modeling languages such as the Entity-Relationship (ER) model and UML are used.

Conceptual database design is *top-down design* as you start by specifying entities (real-world objects) then build up the model by defining new entities, attributes, and relationships.

- ⇨ We will also see a bottom-up design technique called *normalization* where you define the attributes first and then use dependency information to group them into relations.

Database Design
Logical Database Design

Logical database design is the process of constructing a model of the information in the domain using a particular data model, but independent of the DBMS.

The conceptual model is transformed into a logical model such as the relational model.

- It may also be transformed to other logical models such as the object-oriented model, graphs, JSON, or XML.

Since logical design selects a data model, it is now possible to model the information using the features of that model.

- For example, modeling using the relational model allows you to specify keys, domains, and foreign key constraints.

Database Design
Physical Database Design

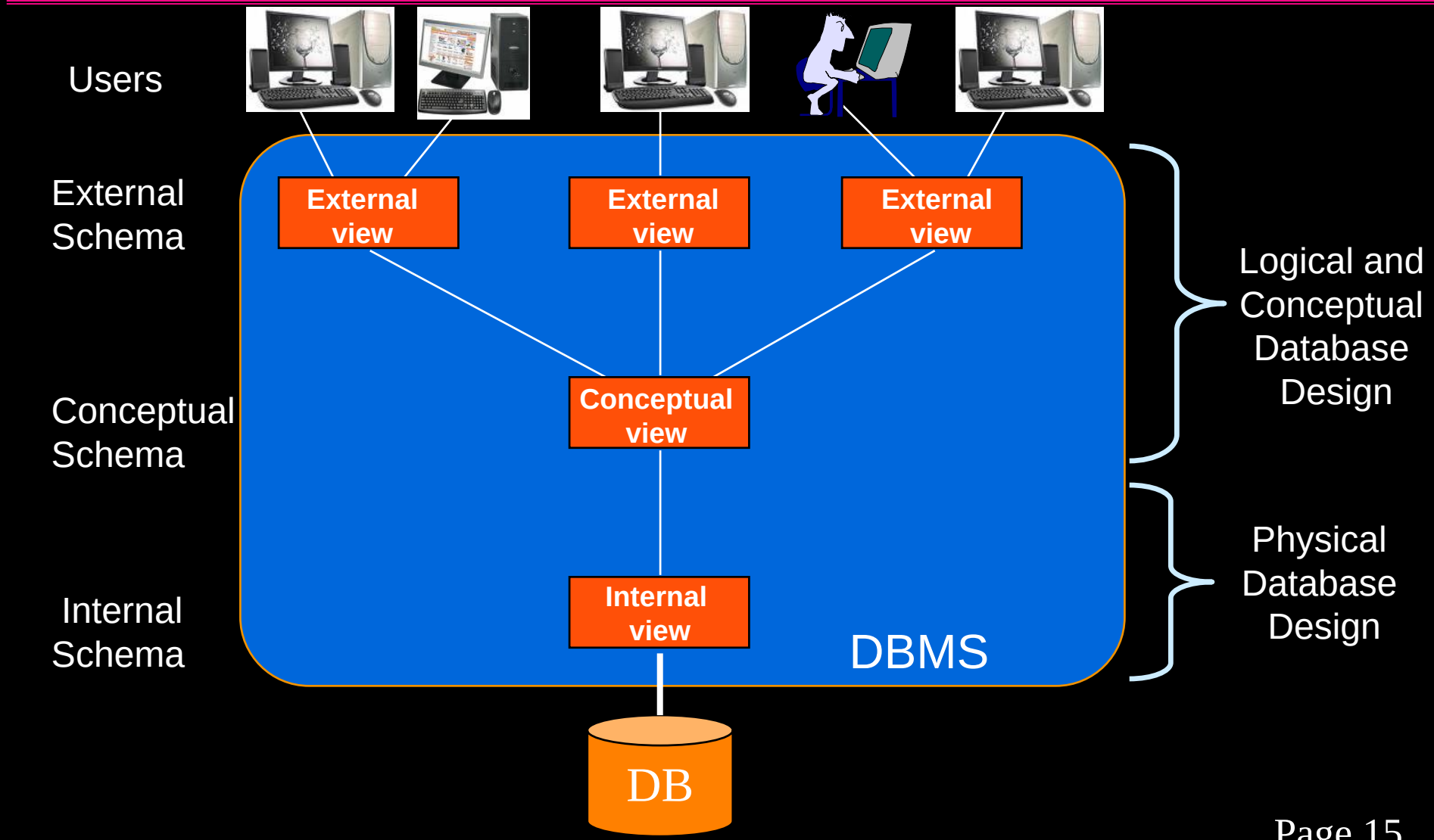
Physical database design is the process of constructing a physical model of information on the secondary storage in a given data model for a particular DBMS.

Physical database design involves the selection of a database system and determining how to represent the logical model on that DBMS. For the relational model this includes:

- creating a set of relational tables using the logical model
- increasing performance using indexes
- implementing constraints and security restrictions
- data partitioning and distribution

Physical database design is *how*, and logical database design is the *what*.

Database Design and the ANSI/SPARC Architecture



Physical Database Design
Selecting a DBMS

The process of a selecting a DBMS is not always simple if the organization does not already have one or there are multiple DBMSs that can be selected.

There are several factors to consider when selecting a DBMS:

- **Features**: Does the DBMS have what you need?
  - Security, constraints, indexes, triggers, SQL-compliance, JDBC/ODBC
- **Compatibility**: Does the DBMS run on your hardware/software?
  - Can the DBMS interface with existing systems?
- **Performance**: Does the DBMS handle the data/query load?
  - **Scalability**: Will the DBMS scale if the data/query load grows?
- **Price**: Does the DBMS fit the project budget?
  - Free is not necessarily cheap. How will the DBMS be supported?

Application Design

The application can be designed *partially* in parallel with the database design.

During the requirements gathering phase, you can collect information on how the data will be manipulated, the general layout of the screens, and requirements that the application will need from the DBMS.

Two main areas of application design:

- **Transactions** - What queries/updates will the application perform?
- **User interface** - How will the data be displayed on the screen and how can the user initiate transactions?

Application Design Recommendations

1) Design the logical database schema first before seriously starting the application design.

- Changes to the logical schema have big impacts on your application and will reduce your development efficiency.

2) Prototype your application, but not your database design.

- It will reduce your development time if you spend the time getting your database design right before building your application.
- Flush out as many details as possible before finalizing your DB design and implementing on the DBMS. Then implement the logical design.
- Prototype your application in stages to get user feedback on the forms and reports as these will frequently change.

3) Isolate database access code into a few classes/methods.

Implementation

The *implementation* stage builds the database and application designs.

The database designs are implemented in the relational model using SQL DDL.

The application is designed in a programming language such as HTML/JavaScript, PHP, C/C++, or Java.

Testing and Maintenance

Testing and *maintenance* are the two final stages:

- ***Testing*** is the process of executing the application programs with the intent of finding errors.
- ***Operational maintenance*** is the process of monitoring and maintaining the system following installation.

There is a significant cost associated with testing and maintaining existing systems. Good design should allow for a system to accommodate changes with minimal reprogramming.

Maintenance also includes monitoring the DBMS for performance, security violations, and upgrades.

CASE Tools

Computer-Aided Software Engineering (CASE) tools can make the development easier.

There are design tools that:

- Automatically convert a high-level design (UML diagram) entered graphically into SQL DDL.
- Examine queries for performance and suggest improvements.
- Provide sophisticated application development environments.

Many DBMSs have monitoring and maintenance systems that assist the DBA in detecting performance issues.

- Called "tuning" a database.
  - An active area of research is the development of systems that tune themselves!

Database People
DA and DBA

We have seen these two database people before:

- ***Database administrator*** (***DBA***) - responsible for installing, maintaining, and configuring the DBMS software.
- ***Data administrator*** (***DA***) - responsible for organizational policies on data creation, security, and planning.

The DA is involved in the early phases of design including planning the project and conceptual and logical design.

The DBA performs the physical design and actively manages deployed, production systems.

Another common position is a (data) ***architect*** that selects systems to use, makes design decisions, and evaluates current and future systems at the architectural level.

Conclusion

Database design is critical to the success of database development projects.

Database design is divided into three phases:

- Conceptual database design
- Logical database design
- Physical database design

Effective requirements gathering is an important skill to master.

Projects should have a well-defined mission statement and project champion to increase their probability of success.

Objectives

- Draw and describe the database development lifecycle.
- Describe the three different steps in database design including the results of each step.
- Describe how the roles of DBA and DA fit into database design. What do these people do?
- List some of the factors to consider when selecting a DBMS for a project.