Fundamentals of Database Systems

Lesson 3: Database Development Process

UNIT OBJECTIVE

After studying this course, you should be able to:

a. Describe the key points of the waterfall model applied to database development
b. Appreciate the roles of various development artefacts, such as the data requirements document, conceptual data model and such like used to communicate between activities in the database development life cycle
c. Understand 3 tierschema/architecture
d. Define: Data independence, instance and schema

Introduction

Database development normally occurs within the context of information systems development. Information systems development is a key organisational process for many organisations (Beynon-Davies, 2002).

Desirable Properties of a Database

Completeness	Ensures that users can access the data they want. Note that this includes **ad hoc queries**, which would not be explicitly given as part of a statement of data requirements.
Integrity	Ensures that data is both consistent (no contradictory data) and correct (no invalid data), and ensures that users trust the database.
Flexibility	Ensures that a database can evolve (without requiring excessive effort) to satisfy changing user requirements.
Efficiency	Ensures that users do not have unduly long response times when accessing data.
Usability (ease of use)	Ensures that data can be accessed and manipulated in ways which match user requirements.

Database Development Life Cycle

Database development is just one part of the much wider field of software engineering, the process of developing and maintaining software.

SDLC and DBLC

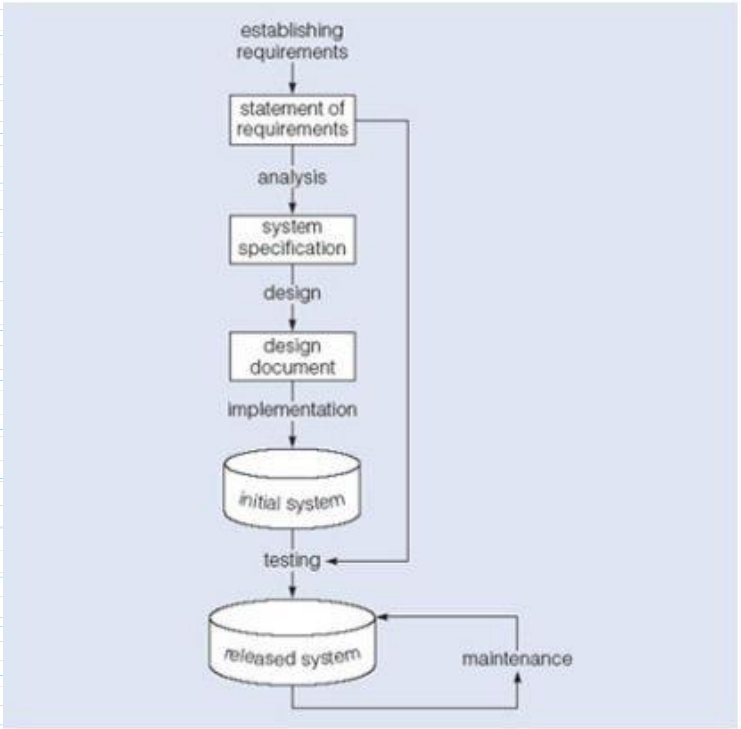


Figure 4 A general model of system development: the waterfall model

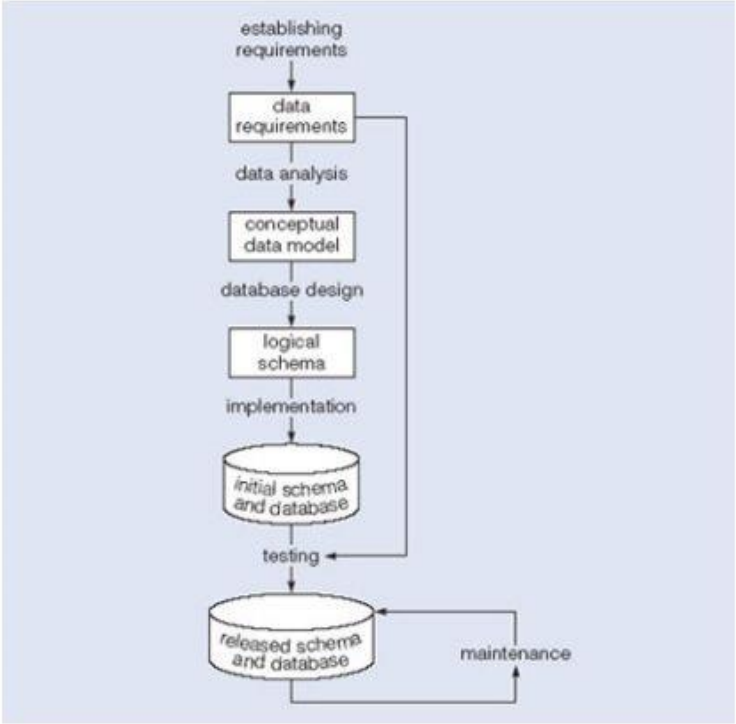


Figure 5 Model of database development

3 assumptions based on the model

- We can separate the development of a database – that is, specification and creation of a schema to define data in a database – from the user processes that make use of the database.

3 assumptions based on the model

- We can use the *three-schema architecture* as a basis for distinguishing the activities associated with a schema.

3 assumptions based on the model

- We can represent the *constraints* to enforce the semantics of the data once, within a database, rather than within every user process that uses the data.

Requirements Gathering

- The data requirements document is used to agree requirements with users.
- The document should give a concise summary of all users' requirements – not just a collection of individuals' requirements – as the intention is to develop a single shared database.

Example

Each course which is available for study is given a course code, a title and a value for credit points – either a 30-point course or a 60-point course. A course may have a quota – the maximum number of students that can be enrolled on the course in any one year; each quota is reviewed regularly and the last date of review is recorded with the quota. A course need not (yet) have any students enrolled on it. Students may not enrol for more than 180 points' worth of courses at any one time.

Analysis

- ***Data analysis*** begins with the statement of data requirements and then produces a *conceptual data model.*
- The aim of analysis is to obtain a detailed description of the data that will suit user requirements so that both high and low level properties of data and their use are dealt with.

Analysis

- *A conceptual data model is concerned with the meaning and structure of data, but not with the details affecting how they are implemented.*
- *The conceptual data model then is a formal representation of what data a database should contain and the constraints the data must satisfy.*

Analysis

- *Analysis focuses on ‘What is required?’ not ‘How is it achieved?’*
- *The aim of analysis is to obtain a detailed description of the data that will suit user requirements so that both high and low level properties of data and their use are dealt with.*

Analysis

The following are the steps in the Analysis Phase.

1. Analyze the organization
2. Define any problems, possibilities or constraints
3. Define the objectives
4. Agree on the scope

Example

- ***In our previous example***

Course Code

Course Title

Credit Points

Exercise

For each of the following statements decide which processes – requirements gathering or data analysis – would generate the statement as part of the documented output.

1. A customer record will allow for the storage of a name, UK address, evening and daytime phone numbers, one mobile phone number and as many email addresses as the customer wants to include.
2. We need to relate customer orders to their credit card details. If the credit card is invalid we need to know before any orders are accepted.
3. An order must have the opportunity to include a delivery address that is different from the customer's credit card billing address.

Exercise

For each of the following statements decide which processes – requirements gathering or data analysis – would generate the statement as part of the documented output.

1. A customer record will allow for the storage of a name, UK address, evening and daytime phone numbers, plus mobile phone number and as many email addresses as the customer wants to include.
2. We need to relate customer orders to their credit card details. If the credit card is invalid we need to know before any orders are accepted.
3. An order must have the opportunity to include a delivery address that is different from the customer's credit card billing address.

Data analysis

Exercise

For each of the following statements decide which processes – requirements gathering or data analysis – would generate the statement as part of the documented output.

1. A customer record will allow for the storage of a name, UK address, evening and daytime phone numbers, plus mobile phone number and as many email addresses as the customer wants to include.
2. We need to relate customer orders to their credit card details. If the credit card is invalid we need to know before any orders are accepted.
3. An order must have the opportunity to include a delivery address that is different from the customer's credit card billing address.

Data analysis

Requirements gathering

Exercise

For each of the following statements decide which processes – requirements gathering or data analysis – would generate the statement as part of the documented output.

1. A customer record will allow for the storage of a name, UK address, evening and daytime phone numbers plus mobile phone number and as many email addresses as the customer wants to include.
2. We need to relate customer orders to their credit card details. If the credit card is invalid we need to know before any orders are accepted.
3. An order must have the opportunity to include a delivery address that is different from the customer's credit card billing address.

Data analysis

Requirements gathering

Requirements gathering

Design

- *Database design starts with a conceptual data model and produces a specification of a logical schema; this will usually determine the specific type of database system (network, relational, object-oriented) that is required, but not the detailed implementation of that design (which will depend on the operating environment for the database such as the specific DBMS available).*

Design

- *The output of the design stage is a detailed relational specification, the logical schema, of all the tables and constraints needed to satisfy the description of the data in the conceptual data model.*

3 General Points basis for Design

- *First, for a given conceptual data model it is not necessary that all the user requirements it represents have to be satisfied by a single database.*
- *Second, remember that one of the assumptions about our database development is that we can separate the development of a database from the development of user processes that make use of it.*

3 General Points basis for Design

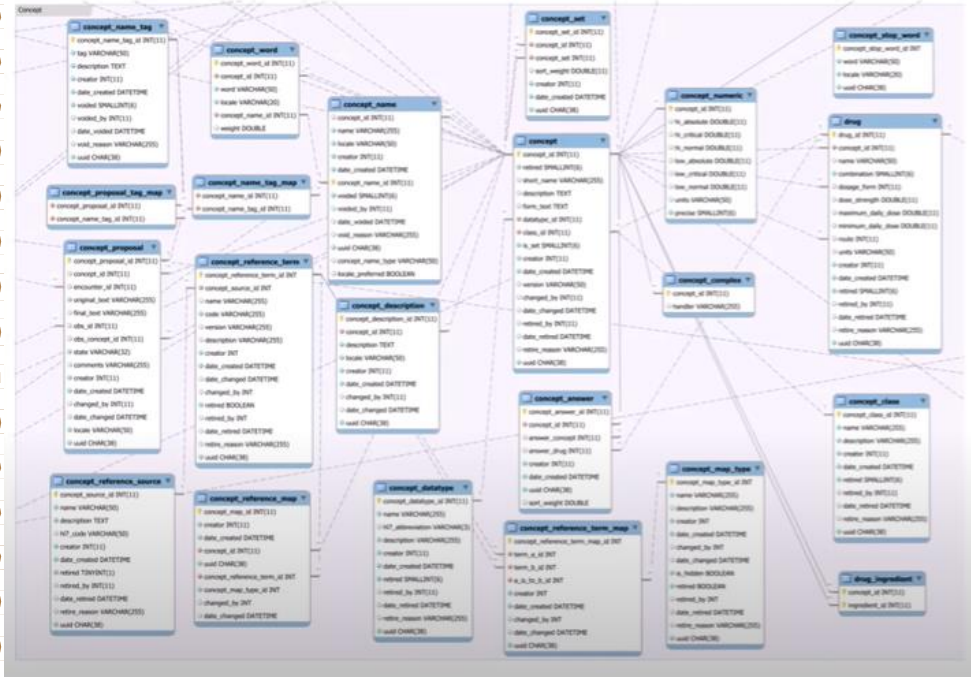
- *Third, at a detailed level, many aspects of database design and implementation depend on the particular DBMS being used.*

Design

This phase consists of three parts: the **conceptual design**, the **logical design** and the **physical design.**

a. **Conceptual** - The purpose of the conceptual design phase is to build a conceptual model based upon the previously identified requirements, but closer to the final physical model. A commonly-used conceptual model is called an entity-relationship model.

What is Data Model?



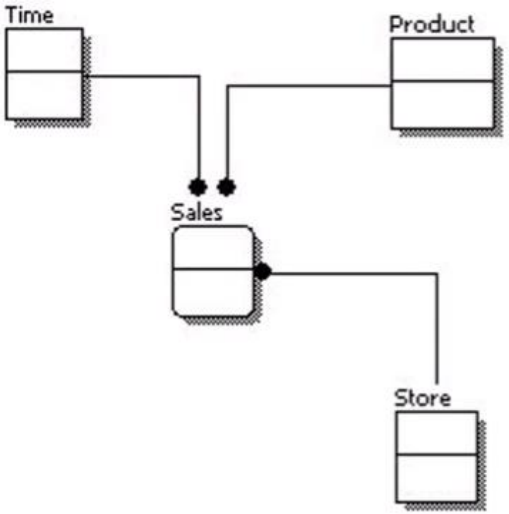
- Pictorial representation of tables
- Represents relationships between tables
- Easily understood

Stages of Data Model

- Conceptual Data Model
- Logical Data Model
- Physical Data Model

Source: Data Academy India
https://www.youtube.com/watch?v=RJ9TpkWKyU0

Conceptual Data Model



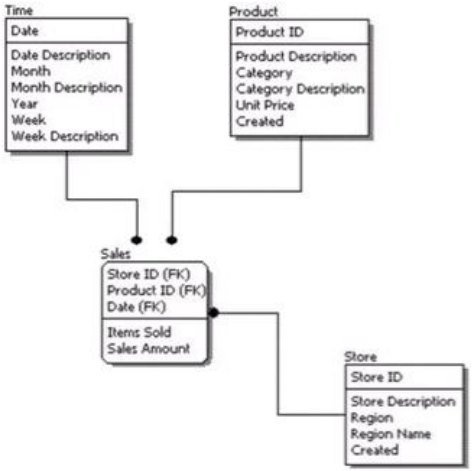
- Highly abstract
- Easily understood
- Easily enhanced
- Only “Entities” visible
- Abstract Relationships
- No software tool is required to define a Conceptual Data Model

Design

This phase consists of three parts: the **conceptual design**, the **logical design** and the **physical design.**

b. Logical - defines HOW the system should be implemented regardless of the DBMS. This phase is typically created by Data Architects and Business Analysts. The purpose is to developed technical map of rules and data structures.

Logical Data Model



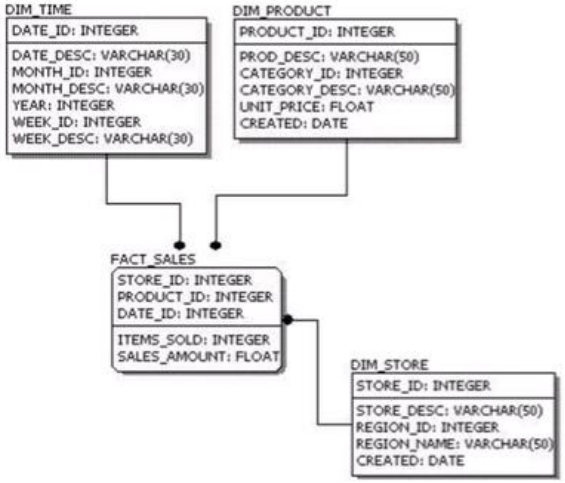
- Presence of Attributes for each Entity
- Key Attributes
- Non-Key Attributes
- Primary Key – Foreign Key Relationships
- User Friendly Attribute names
- More detailed than Conceptual Model
- Database agnostic
- Bit more effort required to enhance, in comparison to Conceptual Model
- Data Modeling tools like ERWin or PowerDesigner can be used to create Logical Data Models. This can be automatically converted to a Physical Data Model with the help of these tools.

Design

This phase consists of three parts: the **conceptual design**, the **logical design** and the **physical design.**

c. Physical - describes HOW the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database.

Physical Data Model



- Entities referred to as Tables
- Attributes referred to as Columns
- Database compatible table names
- Database compatible column names
- Database specific data types
- Difficult for users to understand
- Significantly more effort required to enhance in comparison to Logical Model
- Will include indexes, constraints, triggers & other DB objects
- Difficult to port to a different database, once design is finalized.
- Tools like ERWin and PowerDesigner can help in automatically porting over the Logical Data Model to Physical Data Models of different versions.

Implementation

- *Implementation involves the construction of a database according to the specification of a logical schema. This will include the specification of an appropriate storage schema, security enforcement, external schema, and so on. Implementation is heavily influenced by the choice of available DBMS, database tools and operating environment.*

Implementation

The following are steps in the implementation phase:

1. Install the DBMS.
2. Tune the setup variables according to the hardware, software and usage conditions.
3. Create the database and tables.
4. Load the data.
5. Set up the users and security.
6. Implement the backup regime.

Testing

- The aim of testing is to uncover errors in the design and implementation of the database, its structure, constraints and associated user and management support.
- Testing is usually considered to involve two main tasks – ***validation*** and ***verification***.

Testing

- Validation answers the question: has the right database been developed to meet the requirements?
- It attempts to confirm that the right database has been constructed, with the right characteristics to meet the specified requirements.

Testing

- Verification answers the question: has the database design been implemented correctly?
- Verification ensures that the processing steps, constraints and other ‘programmed’ components of the database (security, backup, recovery, audit trails, etc.) have been correctly implemented and contain no errors in program logic or execution sequences.

Testing

The following are the steps in the testing phase:

1. Test the performance
2. Test the security
3. Test the data integrity
4. Fine-tune the parameters or modify the logical or physical designs in response to the tests.

Maintenance

- Databases are one of the more enduring software engineering artefacts; it is not uncommon to find database implementations whose use can be traced back for 15 years or more.
- Consequently, maintenance of the database is a key issue.

Maintenance

3 Forms of Maintenance:

- **Operational maintenance**, where the performance of the database is monitored.
- **Porting and implementation maintenance**, in which the DBMS, the user processes, the underlying computer system or some other aspect undergoes changes that require the database implementation to be revised.

Maintenance

3 Forms of Maintenance:

- **Requirements change,** where the original requirement specification changes, usually because databases are frequently used for purposes for which they were not originally designed.

Maintenance

The following are the steps in the maintenance phase:

1. Maintain the indexes
2. Maintain the tables
3. Maintain the users
4. Change passwords
5. Backup
6. Restore backups
7. Change the design to meet new requirements

3 Tier Architecture

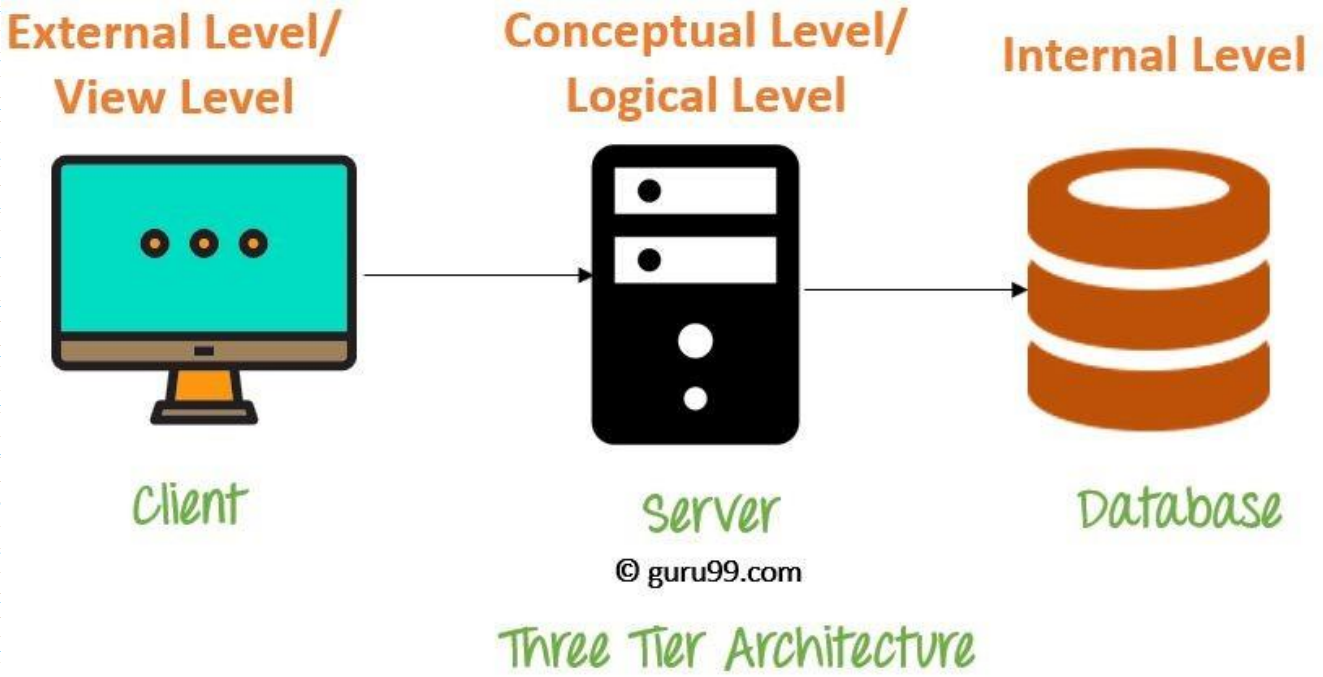
- The main objective of this architecture is to have an effective separation between the *user interface* and the *physical database*. So, the user never has to be concerned regarding the internal storage of the database and it has a simplified interaction with the database system.

3 Tier Architecture

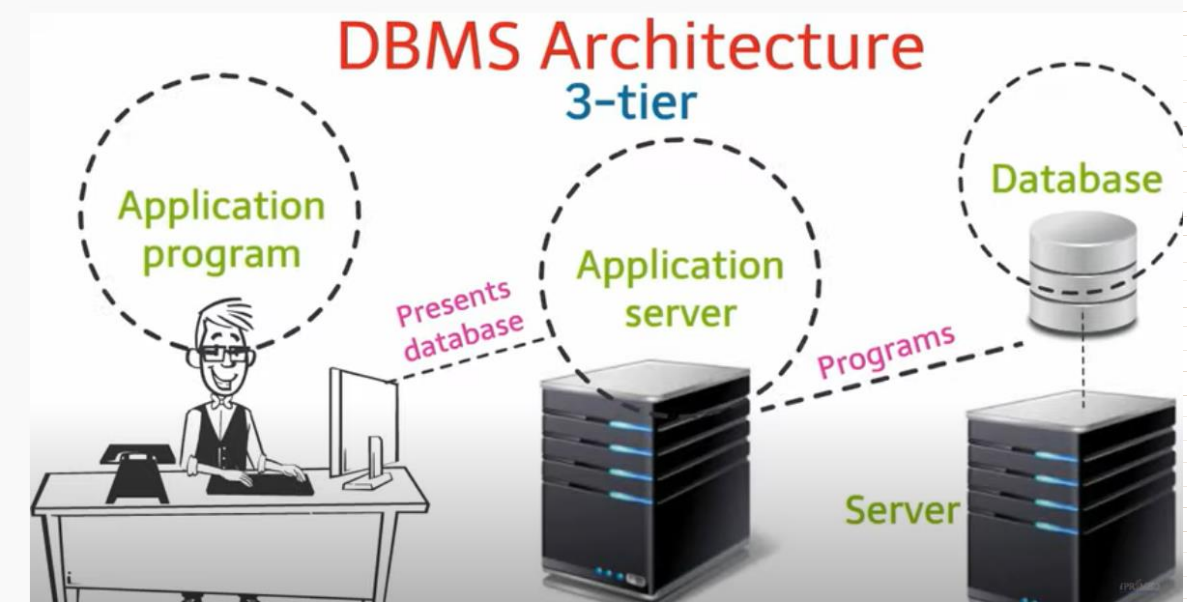
The three-schema architecture defines the view of data at three levels:

- Physical level (internal level)
- Logical level (conceptual level)
- View level (external level)

3 Tier Architecture



Database Architecture



Iprimed Athena
https://www.youtube.com/watch?v=W6P58yb-edE

DBMS Architecture

The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

1 Tier Architecture

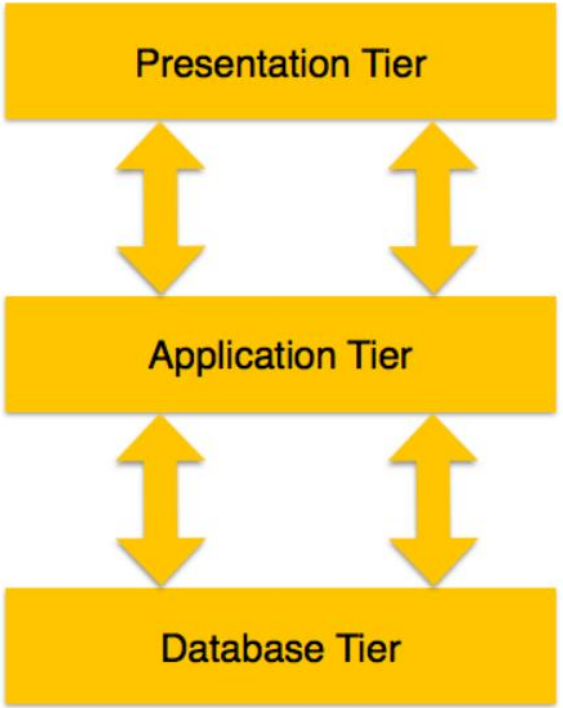
In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

2 Tier Architecture

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.

3 Tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.

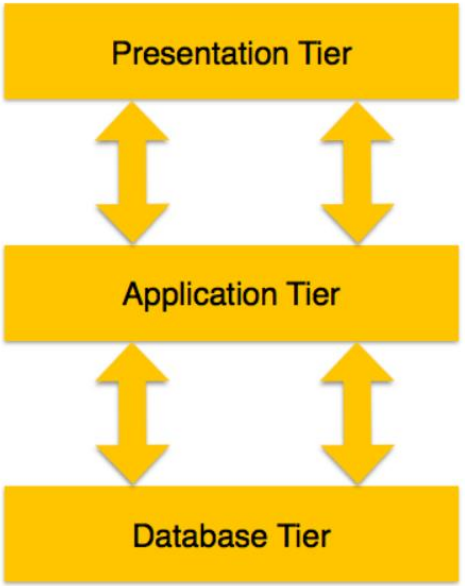


3 Tier Architecture

•**Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.

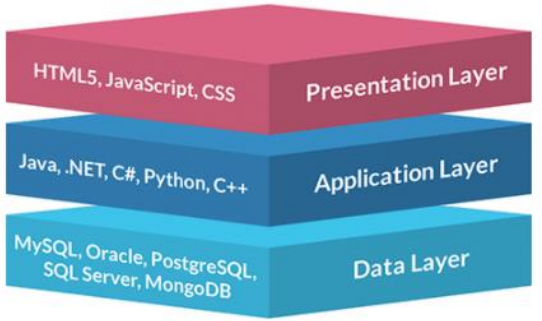
•**Application (Middle) Tier** − At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.

•**User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.



What Do the 3 Tiers Mean?

- **Presentation Tier-** The presentation tier is the front end layer in the 3-tier system and consists of the user interface. This user interface is often a graphical one accessible through a web browser or web-based application and which displays content and information useful to an end user. This tier is often built on web technologies such as HTML5, JavaScript, CSS, or through other popular web development frameworks, and communicates with others layers through API calls.
- **Application Tier-** The application tier contains the functional business logic which drives an application's core capabilities. It's often written in Java, .NET, C#, Python, C++, etc.
- **Data Tier-** The data tier comprises of the database/data storage system and data access layer. Examples of such systems are MySQL, Oracle, PostgreSQL, Microsoft SQL Server, MongoDB, etc. Data is accessed by the application layer via API calls.

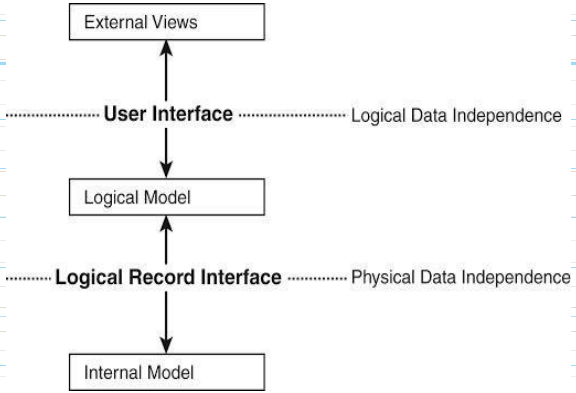


Data Independence

Data independence defines the extent to which the data schema can be changed at one level without modifying the data schema at the next level.

Can be classified into (2) two:

- Logical Data Independence
- Physical Data Independence



Data Independence

- ***Logical Data Independence -*** describes the degree up to which the logical or conceptual schema can be changed without modifying the external schema.
- Changes to data schema at the logical level are made either to *enlarge* or *reduce* the database by adding or deleting more entities, entity sets, or changing the constraints on data.

Data Independence

- ***Physical Data Independence*** **-** defines the extent up to which the data schema can be changed at the physical or internal level without modifying the data schema at logical and view level.
- Physical schema is changed if we add additional storage to the system or we reorganize some files to enhance the retrieval speed of the records.

Instance

What is an instance?

An instance is the retrieval of information from the database at a certain point of time. An instance in a database keeps on changing with time.

Any questions?