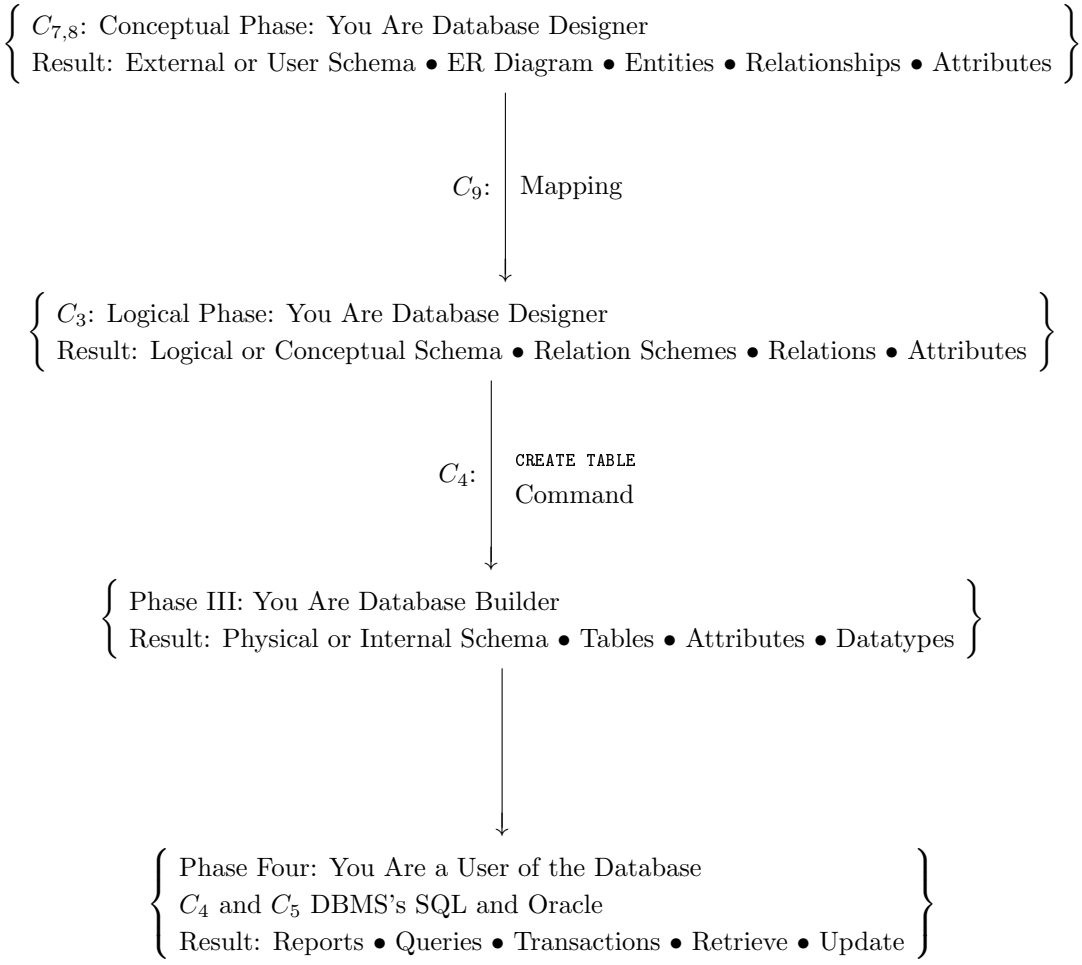
Database Systems

Concepts and Architecture

a. Phases of Database Development. Levels of Data Abstraction. Three Schema Architecture.



1. **Terminology.**

   b. ***Elements of Database Functionality.***

      1. the objects used in the implementation of the programs used to access it,
      2. its self-describing metadata,
      3. the actual data that populates it.

   c. ***Data Model.*** A data model is a collection of concepts such as entities, data types, relationships, constraints, basic operations such as retrieve and update as well as behavioral and dynamic operations, that can be used to describe the structure of a database. There is a data model for each level of user. High level or conceptual for end-users ranging to low level or physical data models for computer specialists. Entities, attributes and relationships are the describing concepts of a conceptual data model. Intermediate or representational data models include network, hierarchical and relational models and they are described as record based. The describing concepts of a low level or physical data model are record formats, record ordering and access paths.

   d. ***Data Abstraction.*** The following three elements of database design, the objects used in the implementation of the programs used to access it, its self-describing metadata, the actual data that populates it, are stored and maintained separately and independently so that a change to one does not necessitate a change to the others. Any user of a database is generally primarily concerned with one of these three elements of database design. A database supports various kinds or levels of users, casual to professional end-users, administration and maintenance personnel who control access to and maintain the database, database designers who have to understand end-user requirements and develop the structures needed to satisfy their needs, application programmers who create programs that implement end-user specifications. Data abstraction refers to the development and maintenance if several descriptions of data that focuses on the features and requirements essential to each kind of user and each level of use. In short, the typical end-user is not concerned with the details of data storage.

   e. ***Data Independence.*** Data independence requires that a change at any one level of data representation be invisible to the other levels; in other words, a change at one level should not necessitate a change at the other two level. When data independence is enforced at all levels, a programmer working at one level does not have to be concerned with changes that are taking place at the other levels.

   f. ***Database Schema.*** A database schema is a description of the database, not the data itself. A schema diagram includes names of entities and attributes of each. This information is kept in the catalog of the DBMS and is also called metadata.

   g. ***Database State.*** A database state or snapshot is the data, the current set of instances, records or tuples, in a database at a particular time. The database schema contains the definition of a database and the database state contains a populated database.

   h. ***Three Schema Architecture.*** This refers to three different descriptions of the data of a database used to meet the requirements and understanding of its various kinds and levels of users. This architecture is aimed at achieving and visualizing three of four main characteristics of the database approach.

      1. **Characteristics of the Database Approach.**

         a. **Self-Describing.** The catalog of metadata which describes the database and is maintained to by the database.

         b. **Program / Data & Program / Operation Insulation or Independence.** The following three elements of database design, the objects used in the implementation of the programs used to access it, its self-describing metadata, the actual data that populates it, are stored and maintained separately and independently so that a change to one does not necessitate a change to the others.

         c. **Support of Multiple User Views.** This refers to the multiple virtual (non-permanent) views that a database can provide to satisfying the requirements, which come in the form of queries, of its various users.

         d. **Sharing of Data and Multiuser Transaction Processing.**

2. **Levels of Database Architecture.**
    a. **Internal or Physical Schema.** SDL.
    b. **Logical Schema.** DDL.
    b. **Conceptual or External Schema.** VDL and DML.
3. **DBMS Architecture.**
    a. **Centralized DBMS Architecture.**
    b. **Client/Server DBMS Architecture.** Any network consists of specialized servers which make resources available to clients machines and their users. Clint machines provide their users with interface capabilities and local processing. File Server. Printer Server. DBMS Server. All print requests are forwarded to the printer server which organizes, prioritizes and executes them. Web Server. Email Server.
    c. **Two-Tier Client/Server DBMS Architecture.** In this architecture the components of a DBMS are distributed as follow.
        1. ***Server Side.*** The DBMS. For an RDBMS, the SQL is the query processing language and the server is called the SQL server or the query server.
        2. ***Client Side.*** User interface and application programs.

        Client/server interactions are controlled by users and their programs. Using an ODBC API (application program interface meeting the open database connectivity standard), a client program can connect to one or more RDBMS servers and send query and transaction requests, request are processed by the server and results are returned to the client where they are collected and displayed.

        Alternative DBMS-Integrated Two-Tier Architecture. DBMS modules are distributes between client and server sites and client / server interaction is controlled by yet other DBMS modules, these also being held in either client or server sites.
        1. ***Server Side.*** SDL software for data storage on disk pages, local concurrency control and recovery, buffering and caching of disk pages, and so on.
        2. ***Client Side.*** user interface, data dictionary functions,DBMS interactions with programming language compilers, global query optimization, concurrency control and recovery across multiple servers, structuring of complex objects from the data in the buffers and so on.
    d. **Three-Tier and N-Tier Architectures for Web Applications.**
        1. ***Client Tier. User Interface Tier.*** This tier include a GUI succh as a web browser and application dependent business rules. This tier accepts data entry from the user and displays information and responses from the other tiers to the user.
        2. ***Application or Web Server. Business Logic Layer.*** This intermediate tier stores application programs and business rules including software to perform inspections of client credentials [passwords and credit card numbers]. This tier verifies business rules and performs security inspections before data is passed down to the database server or up to the user. This is were the web server resides in web-based applications. The web-server accepts results from the database server and formats them into web pages that are viewed by the user through the web browser
        3. ***Database Server Tier. Data Access Tier.*** This layer stores and executes all data management services.

        The intermediate server accepts requests from the client, processes requests and sends database queries and commands to the database server then acts as a conduit for passing partially processed data from the database server to the client where it is further processed and filtered for presentation to users in the gui format.

        In N-tier architecture, business logic is distributed among additional tiers.

i. Query Language.

j. ***Host Language.***

k. ***Data Sublanguage.***

l. ***Database Utility.***

The Three-Schema Architecture of Database Design	\dbtables\threelevels
Conceptual Database Design Phase I High Level User Level What? External Schema View Definition Language (VDL) Data Manipulation Language (DML)	This includes investigation into user requirements and an elucidation of the data requirements to meet their requirements. Data at this level is viewed as it is perceived by expected user's of the database. The collection of methods and constructs used at this level to describe the data are called the external schema. The result of this phase is a conceptual data model which is summarized by an E/R diagram. The ER diagram should describe the views of the database of its prototypical users. A portion of the E/R diagram of a typical business is this, $\boxed{\text{Customers}} \xrightarrow{\text{(many)}} \langle Place \rangle \xrightarrow{\text{(one)}} \boxed{\text{Orders}}$ Entities are indicated by [] and relationships by $\langle\rangle$. Both Customers and Orders are represented by tables, each having a key, customer id and order id respectively. The Place is a table with attributes the customer id and order id combining to make up its key. Place is a dependency of the other two tables. The system won't allow a change to the customer id in the presence of a dependency. A VDL is language used to specify the external schema (ER diagrams) of a database. A DML is the language used by user's of the database to retrieve, insert, delete and modify the database. The DML creates views, also called virtual or derived tables, of the data. A high level DML is declarative in that the user describes what he\she wants without concern as to how its accomplished. A query in the relational calculus for instance, because it involves a sequence of operations, is considered to be too complicated for the typical commercial DBMS user because the sequencing of operations involves details of how the query is to be carried out. The high level declarative interface takes care of such details. Many records can be retrieved with a single statement from a high level DML. Statement of a high level DML can be embedded in the code of a general purpose language. A query language is a high level DML used i a stand alone interactive manner. A low level DML can access only one record at a time and its statements must be embedded in a general purpose (host) language such as COBOL.

Logical Database Design
Phase II
Mid Level
How?
Logical Schema
Data Definition Language (DDL)
	Conceptual data model $\longrightarrow$ Logical data model or a schema which includes:
	• A logical data model is described by one of these terms $\{$relational, object-oriented, hierarchical, network$\}$ and one of these terms $\{$implementation, representational$\}$. The data model describes data types of the attributes of the entities (objects) and relationships and constraints among the attributes of each entity. It also provides operations on the data including retrieval and updating.
	• The construction of a database catalog. The catalog contains information, collectively called the database schema, that describes the data of the database and such information, data describing other data, is called metadata. Though the data it describes is likely to change regularly and often, metadata rarely changes. Metadata includes schema constructs which describe the objects or entities of the database. Metadata in its final form consist of the names of the tables of the database along side the attributes of each table displayed as column headers of the table. It also includes the following information.
	  ◦ database name, owner, description, creator, status(test or production), copy or version number.
	  ◦ table name, owner, description, creator, synonyms, table column synonyms, data sources, cross-references, related columns.
	• normalization which is a optimization process.
	An instance or database state is a snapshot of the database taken at a specified time. It consist of the tables of the database as they are populated at the specified time. The database might be called the intention of the database and a state of the database might be called an extension of the database schema.

Physical Database Design Phase III Low Level Machine Level How? Physical Schema Storage Definition Language (SDL)	This level is concerned with details of the way data is stored on the computer, data structure types and implementations such as linked lists, B-trees, hash files, access mechanisms, record formats and ordering, and physical requirements and constraints, efficiency and performance of the database. The collection of methods and constructs used at this level to describe the data are called the physical schema.

2. **Data Model.** A collection of concepts that can be used to describe the structure of a database and to facilitate data abstraction. DBMS's use data models to implement databases. The main example of this course is the Entity-Relationship Model, a conceptual model. Among the representational (record-based models) and implementation data models are the relational data models [1] and the legacy, network and hierarchical, models. The object data model is a conceptual implementation data model often used in software engineering.

   a. ***Elements of a Data Model.***

      1. **Structure of a Database.** This is described in terms of data types, relations, and constraints.
      2. **Basic Operations.** These are operations for various kinds of retrieval and updating.
      3. **Dynamic Behavior.** Specifications of valid user-defined operations, the dynamic behavior, of a database application.

   b. ***Categories of Data Models.*** There is a range of data models: high-level or conceptual to low-level or physical data models. An access path is a structure used in low-level models that aim at the efficient implementation of searches for database records. An index is a access path that allows direct access to data.

   c. ***Concepts of a Data Model.***

      1. **Entities.** An object or concept of the model's UoD. Tables of the database.
      2. **Attributes of an Entity.** These are variables representing properties of the model's entities. Headers of the columns of the tables.
      3. **Relationships among Entity.** Two entities are related if they have a common attribute.

   d. ***Data Base Schema. The Description of the Database.*** A schema diagram shows each entity of the database and its attributes displayed as the header row of a table. The schema of a db includes constraints and data types, generally not shown in a schema diagram. Change to the schema, called schema evolution, should be rare whereas snapshots, also called db states or sets of instances, change regularly. The definition of a new db amounts to the empty, unpopulated or unloaded state. Each update brings about a new state and it is up to the DBMS to check that the new state is a valid one, satisfies all the constraints of the db. The DBMS stores the definition of the db, the metadata, it the catalog portion of the db.

3. **Three-Schema Architecture.**

   a. ***External Level Schema.*** User-specific view, hides other such views, details of storage and implementation.

   b. ***Conceptual Level Schema.*** Describes the structure of the database for all users, hiding details of storage and implementation, and concentrating on entities, relationships, operations and constraints.

   c. ***Internal Level Schema.*** describes the physical storage and implementation details of the db.

   In a Three-Schema DBMS, each user request must be translated to the conceptual level and from there to the internal level. Data from the internal level must likewise be transformed (mapped) between levels to match the user's external view.

   Logical and Physical Data Independence. Application programs and the external schema should continue to work without change when changes are made to the conceptual level. The conceptual schema should continue to work without when changes are made to the way data is managed at the internal schema. The transformations between the schema, which are maintained by the DBMS in its catalog area, are expanded to accommodate any changes.

4. **Database Languages For Defining Each of the Three Schema.**

   a. ***View Definition Language. (VDL).*** to implement various user vies. SQL is the VDL for RDBMS's.

   b. ***Data Definition Language. (DDL).*** for implementation of the conceptual schema

   c. ***Storage Definition Language. (SDL).*** for implementation of the internal schema

[1] C_3 is on relational data models, their constraints, operations and languages, C_4 and C_5 are on SQL which is the standard ??? for relational databases.

Even when there is a clear separation between schema, often one language plays a duo role. Also, one of the languages is used to implement transformations between schema.

d. ***Data Manipulation Language. (DML).*** This is the language used to implement manipulations of data, such as retrieval, deletion, insertion, updating, of a defined and populated db.

1. **High Level Nonprocedural DML.** Statement are entered interactively from monitor (A standalone DML is called a query language) or they're embedded in an general purpose programming language (called the host and the DML is the data sublanguage) such as C++. Such statements are identified and extracted by a preprocessor, processed by the DBMS, then processed by the gpl. These languages set-oriented DMLs (eg. SQL retrieves many records with a single statement) and declarative (specifies which data to retrieve not how to retrieve it).
2. **Low Level Procedural DML.** Statements must be embedded in an general purpose programming language such as C++. These languages one record-oriented DMLs and specify which data to retrieve and how to retrieve it.

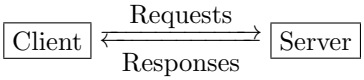
These are not distinct languages of modern DBMSs but their roles are carried out by components of comprehensive integrated language. SQL=VDL+DDL+DML but not SDL. DBA staff uses SDL to refine and optimize db performance.

e. ***DBMS Interfaces.*** Many, probably most, applications, a web browser for instance, are user friendly interfaces between their user and a db.

1. **Menu-Based Interfaces.**
2. **Form-Based Interfaces.**
3. **GUIs.**
4. **Natural Language Interfaces.**
5. **Speech I/O.**
6. **Interfaces for Parametric Users.** User's requiring only a small set of operations.
7. **Interfaces for DBAa.** Permit access to privileged commands, account creation, control of system parameters, granting of special authorization, changing schema, reorganizing the storage structure of the db.

5. **Database System Environment.**
   - *a*. ***Component Modules.***
   - *b*. ***Database System Utilities.***
   - *c*. ***Tools.***
   - *d*. ***Application Environments.***
   - *e*. ***Communication Facilities.***

6. **Criteria for Classification of DBMSs.**
   a. ***Data Model.*** Relational, object (classes are organized into hierarchies), object-relational, legacy (hierarchical (IMS by IBM) and network (IDMS)), XML model (used for exchange of web data, uses object style with different terminology).
      1. **Hierarchical.** data represented in a hierarchical tree structure, each hierarchy representing a number of related records.
      2. **Network.** data represented as record types, one to many (1 record related to many records by pointer mechanism) relationships called set-types
      3. **Relational.**
      4. **Object Oriented.**
      5. **Deductive.**
   b. ***Number of Users.*** Single user (PCs) vs. multiple user (must support concurrency).
   c. ***Distribution of DBMS Functionality.*** DBMS Functions. Centralized (data stored on one computer) vs. distributed DBMS (DDBMS)
      1. **Centralized DBMS Architecture.** One site for all DBMS functions for use by connected terminals without processing power.
      2. **Basic Client/Server Architectures.**



      3. **Centralized Client / Server Architecture of a DBMS.** User interface processing, application program execution, and data processing, are carried out on a single machine.
      4. **2-Tier Client / Server Architecture of a DBMS.**
      5. **3-Tier Client / Server Architecture of a DBMS.**
   d. ***Software Distribution Among Sites.*** Homogeneous (same software at every site), heterogeneous (different software at various sites),
   e. ***Cost.*** Site to single-user licenses.
   f. ***Access Path Options for Storing Files.***
   g. ***Use.*** General purpose vs. specific.
7. **Utilities of a Database System.**
   a. ***Loading.***
   b. ***Backup***
   c. ***Database Storage Organization.***
   d. ***Performance Monitoring.***
   e. ***Report Generation.***
   f. ***Sorting.***
   g. ***Data Compression.***

BIBLIOGRAPHY

[1] *R. Elmasri & S. B. Navathe,* "Fundamental of Database Systems," 6^{th} Ed. Addison Wesley 2011.
[2] *P. Dubois,* "MySQL Developer's Library," 4^{th} Ed. Addison Wesley 2009.
[3] *S. Dietrich,* "Understanding Database Query Language," Prentice Hall (2001).