Chapter1
Introduction

Introduction

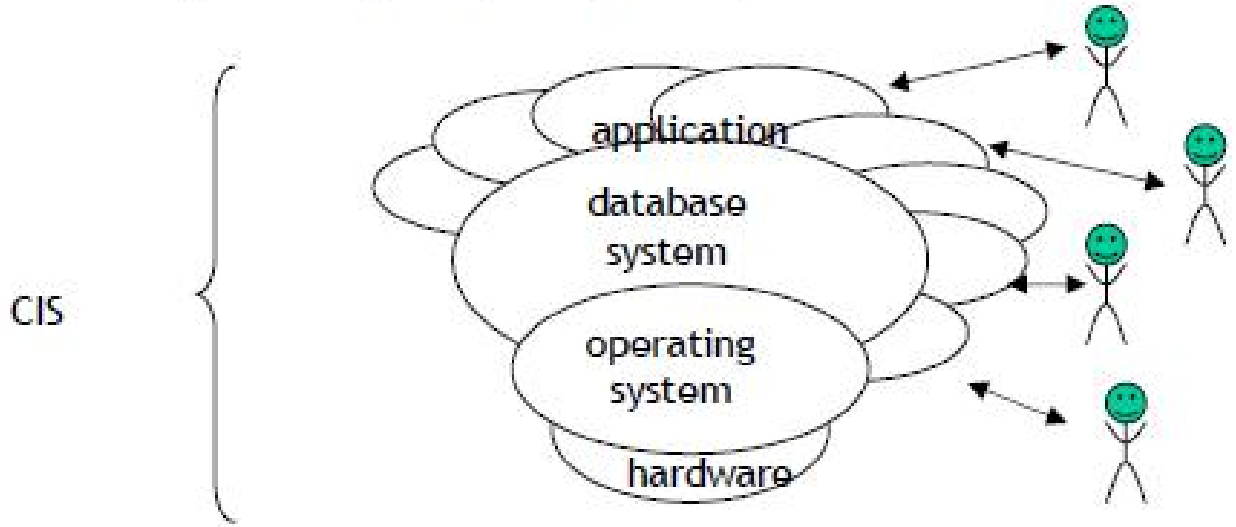
Database is a collection of interrelated data that are used by application. It contains information relevant to an enterprise.

DatabaseManagementSystem(DBMS) is a collection of interrelated data and a set of programs to access those data.

PrimarygoalofDBMS is to provide a way to store and retrieve database information conveniently and efficiently.

Computer-based information system

Database systems (DBS): key component for CIS



DBS = DB + DBMS

Database Management Systems as Part of a CIS

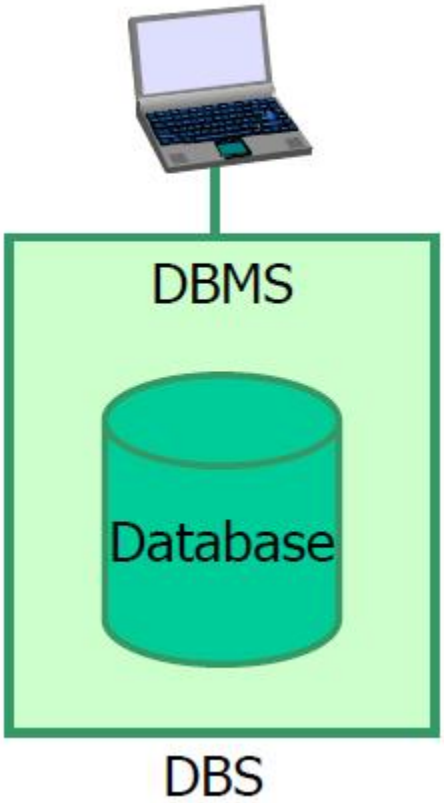
DBMS: A tool for creating and managing large amounts of data efficiently and allowing it to persist over long periods of time, safely. (Garcia-Molina et. al., 2002)

Databases today are essential to every business, in order to:

- present data to customers
- present data on the World-Wide-Web
- support commercial processes

Some important capabilities:

- Persistent storage for large amounts of data
- Programming interface for users and applications
- Transaction management



Purpose of Database System

To see why database management systems are necessary, let's consider typical file-processing system supported by a conventional operating system.

Consider an application in SAVING BANK

- Information of saving account and customer records are kept in permanent system files.
- Application programs are written to manipulate files to perform following tasks
  - Debit or credit an account
  - Add a new account
  - Find an account balance
  - Generate monthly balance

Development of the system proceeds as follows

- SAVING + CURRENT Account Bank
- New application program must be written as needed
- New permanent files are created as required
- But over a long period of time files may be in different format
- Application programs may be in different languages.

Disadvantages of using file-processing system (1)

(Purpose of Database System)

1. Data redundancy and inconsistency
   - Various files are likely to have different format
   - Programs are written in different languages
   - Duplicate information in several places as new CURRENT A/C added
   - All copies may not be updated properly leading to data inconsistency.
   - Redundancy leads to higher storage and access costs.
2. Difficulty in access data
   - May have to write a new application program to satisfy an unusual request.
   - E.g. list all customers with the same postal code.
   - Could generate this data manually, but a long job or new application program is need to be written
   - No convenient and efficient retrieval of data
3. Data isolation
   - Data in different files and in different formats
   - Difficult to write new application programs.

Disadvantages of using file-processing system (2)

(Purpose of Database System)

4. Integrity problem
   - Data may be required to satisfy constraints.
   - E.g. no account balance below $25.00.
   - Again, difficult to enforce or to change constraints with the file-processing approach.
5. Atomicity problems
   - Data need to be restored to the consistent state if a failure occurs.
   - E.g. a program to transfer $50 from account A to account B
   - If a system failure occurs, it is possible that the $50 is removed from account A and is not credited to B, resulting into inconsistent state.
   - It is essential that either both debit and credit occurs or none.
   - Difficult to ensure atomicity in file-processing system.

Disadvantages of using file-processing system (3)

(Purpose of Database System)

6. Concurrent-access anomalies
   - Want concurrency for faster response time.
   - Need protection for concurrent updates.
   - E.g. two customers withdrawing funds from the same account at the same time. Account has $500 in it, and they withdraw $100 and $50. The result could be $350, $400 or $450 if no protection.
7. Security Problem
   - Every user of the system should be able to access only the data they are permitted to see.
   - E.g. payroll people only handle employee records, and cannot see customer accounts; tellers only
   - access account data and cannot see payroll data.
   - Difficult to enforce this with application programs.Security problem

These difficulties have prompted the development of database system.

Benefits of database approach

1. The redundacy can be reduced.
2. The data can be shared.
3. Inconsitency can be avoided.
4. Transaction support can be provided.
5. Integrity can be maintained.
6. Security can be enforced.
7. Conflicting requirements can be balanaced.
8. Standards can be enforced.

View of Data

ViewofData

- Data Abstraction
- Instances and Schemas
- Data Independence

The major purpose of a database system is

- to provide users with an abstract view of the system
- i.e. the system hides certain details of how data is stored and maintained

Data Abstraction

Data abstraction refers to hiding of certain details of how the data are stored and maintaned.

That means hiding complexity of the database system from database users with several levels of abstraction.

Three levels of Data Abstraction

View Level

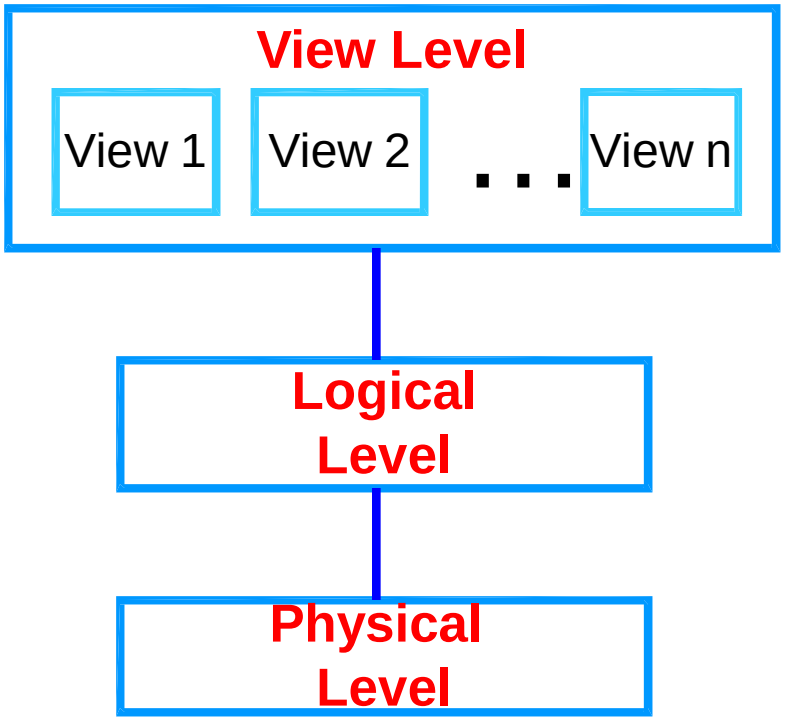
- Highest level of abstraction
- What data users and application programs see ?
- Describes only part of the database for a particular group of users
- Different views of database
- Eg. Tellers in a bank get a views of customer accounts but not payroll data.

Logical Level

- What data are stored in database?
- Relationship between those data.
- User of the logical level does not need to be aware of complex physical level structures.
- Used by database adminsitrators.

Physical Level

- Lowest level of abstraction
- How data are actually stored?
- Complex low-level data structure are described in detail. Eg. Indexing, Hashing



Example of three levels

External (PL/1)

```
DCL   1 EMPP,
        2 EMP# CHAR(6),
        3 SAL FIXED BIN(31)
        4 NAME CHAR(20)
```

External (COBOL)

```
01   EMPC.
       02   EMPNO PIC X(6).
       02   DEPTNO PIC X(4).
```

Conceptual

```
EMPLOYEE
        EMPLOYEE_NUMBER      CHARACTER   (6)
        NAME                 CHARACTER   (20)
        DEPARTMENT_NUMBER    CHARACTER   (4)
        SALARY               NUMERIC     (5)
```

Internal

```
STORED_EMP      LENGTH=40
        PREFIX  TYPE=BYTE(6), OFFSET=0
        EMP     TYPE=BYTE(6), OFFSET=6, INDEX=EMP
        NAME    TYPE=BYTE(20), OFFSET=12
        DEPT#   TYPE=BYTE(4), OFFSET=32
        PAY     TYPE=FULLWORD, OFFSET=36
```

Note: Refer Date, C. J. An introduction to Database Systems

Analogy to Data Types concept

Following code defines a new record type, customer with four fields.

```
//In Pascel Programming
type customer=record
                    customer-id:string;
                    customer-name:string;
                    customer-street:string;
                    customer-city:string;
        end;
```

```
//In C++ Programming
typedef struct customer{
            string customer-id;
            string customer-name;
            string customer-street;
            string customer-city;
}
```

At physical level,

- Block of consecutive storage (Words or bytes)
- The compiler hides the level of details from programmer.

At logical level,

- Records described by a type definition as above and
- interrelationship of these records.
- Programmers and Database adminsitrators.

At View level,

- Computer users see a set of applicaton programs that hides details of the data types.

Instances and Schemas

Database schema

- The overall design of the database is called the database schema.
- The description of a database which is specified during database design and is not expected to change frequently.

Database Instance

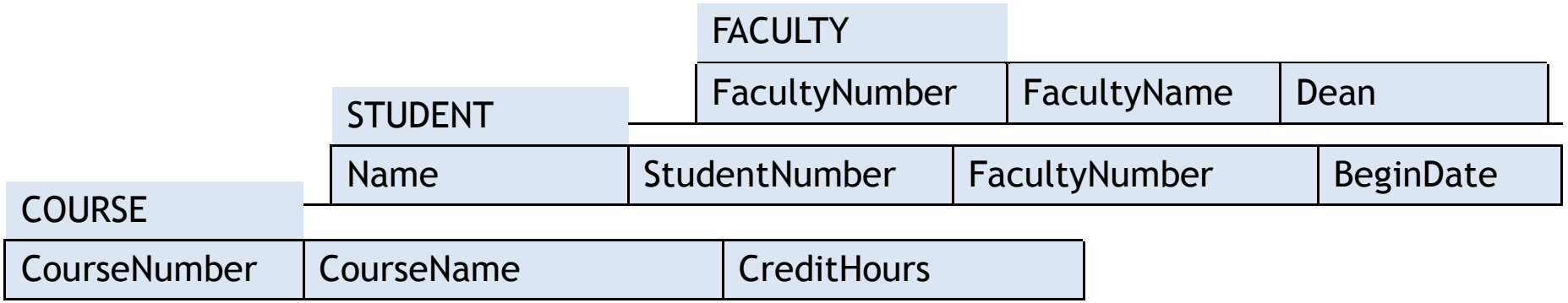
- The collection of information stored in the database at a particular moment in time.

Analogy with programming language:

- Variable declaration with data type – Schema
- Value of a variable at a given instant - instance

Instances and Schemas

SchemaDiagramofDatabase



Instance(State)

FACULTY	FacultyNumber	FacultyName	Dean
	101	Business	1231
	102	Engineering	1232

STUDENT	Name	StudentNumber	FacultyNumber	BeginDate
	Smith	112321	101	01-09-2009
	John	823431	102	01-09-2009

Three-Schema Architecture (1)

(ANSI/SPARC Architecture)

External Schema/Subschema (user's view):

- Describes severals schemas at view level.
- Describes the part of the database, that a particular user group is interested in.

Conceptual Schema (logical system view):

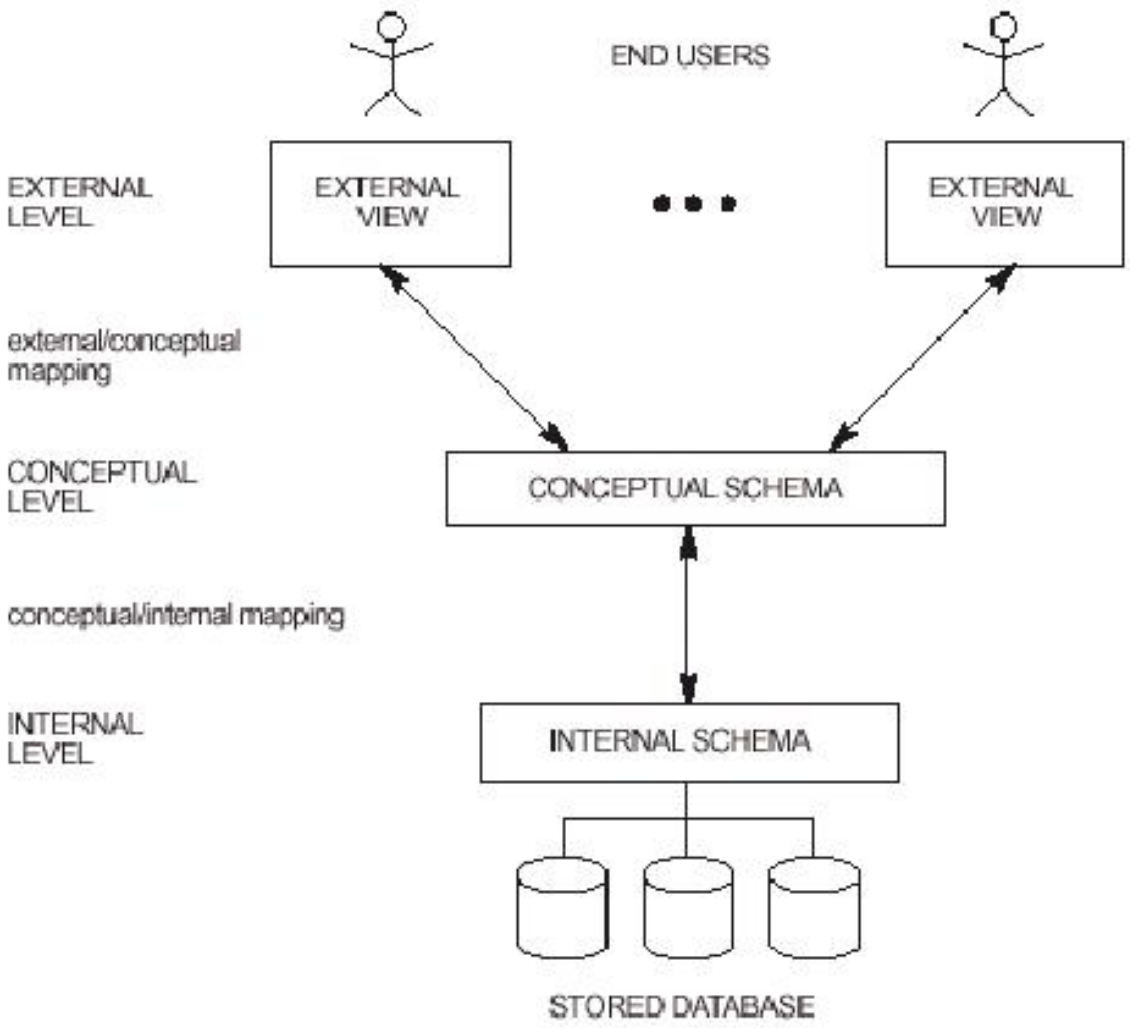
- Describes the database design at logical level.
- Describes the structure of the whole database for a community of users
- Data Definition Language (DDL)

Internal Schema (physical system view):

- Describes the database design at the physical level.
- Determines the physical storage structure of the database (e.g. formats of physical records, access path)
- Storage Structure Language (SSL)

Three-Schema Architecture (2)

(ANSI/SPARC Architecture)



Three-Schema Architecture (3)

(ANSI/SPARC Architecture)

The three schemas are only descriptions of data

The only data that actually exists is at the physical level.

In a DBMS based on the three-schema architecture,

- each user group refers only to its own external schema.
- Hence, the DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database. If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view.
- The processes of transforming requests and results between levels are called mappings.

Data Independence

the capacity to change the schema at one level of database system without having to change the schema at next higher level.

Two types of data independence

LogicalDataIndependence

is the capacity to change the conceptual schema without having to change external schema or application program.

PhysicalDataIndependence

is the capacity to change the internal schema without having to change the conceptual schema (or external schema).

*Modificaton at this level is usually to improve performance.

Logical Data independence (Examples)

Changing conceptual schema by adding or removing record type or data item.

Initially

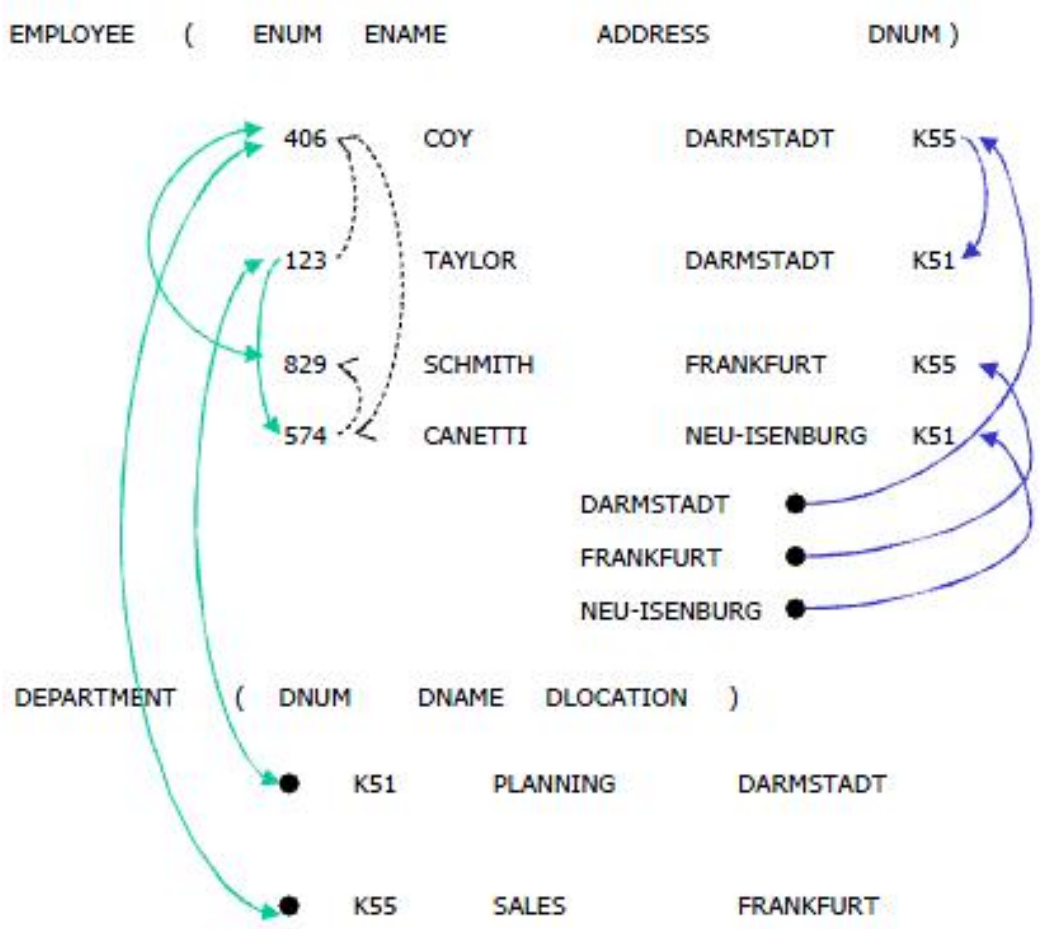
STUDENT	Name	StudentNumber	FacultyNumber	BeginDate
	Smith	112321	101	01-09-2009
	John	823431	102	01-09-2009

AfterremovingFacultyNumber

STUDENT	Name	StudentNumber	BeginDate
	Smith	112321	01-09-2009
	John	823431	01-09-2009

Only the view definition and mapping need to be change in DBMS but not an application program.

Physical Data independence (Examples)



Database Languages (1)

The **DDL (Data Definition Language)**

- is used to define the database schema
- Used by database administrator and database designer

```
create table account(
        account_number char(10);
        balance integer
        );
```

Data Definition (DDL):
CREATE SCHEMA
CREATE DATABASE
CREATE TABLE
CREATE VIEW
ALTER TABLE

- Execution of the DDL states creates account table and updates special set of tables called the data dictionary.
- A data dictionary contains metadata- that is data about data. Eg. Schema of table
- The data dictionary is consulted for reading or modiying actual data.
- The storage structure and access methods used by the database system are specified by a set of definitions in a special type of DDL called a data storage and definition language
- basic idea: hide implementation details of the database schemas from the users
- The DDL also provide facilities to specify contraints. For example the minimum balance Rs. 1000

Database Languages(2)

The DML (Data Manipulation Language) is

- used to retrieve, insert, delete and update data in the database.
- In short access or manipulate data
- used by database users

Two types of Data Manipulation Langauge:

Procedural DMLs

a user is required to specify WHAT data are needed as well as HOW to get those data.

Declarative (Non-Procedural) DMLs

a user is required to specify ONLY WHAT data are needed but NOT HOW to get those.

Database Languages (3)

MoreonDeclarativeDMLs

Database system has to figure out efficient means of accessing data as DML while using Declarative DMLs.

SQL(Structured Query Langauge) language is a non-procedural langauge.

A Query is a statement requesting the retreival of information.

A Query language is the portion of DML, that involves information retrieval

```
--ExampleSQL query to find the name
of the customer whose customer_id=192

select customer.customer_name
from customer
where customer.customer_id=192
```

Database Languages (4)

The DCL (Data Control Language)

- is used to control the access to data in the database.
- Data Control commands:
  - constraints definitions with CREATE TABLE
  - Grant and deny permissions GRANT & REVOKE
  - COMMIT – makes all changes visible to users
  - ROLLBACK – undoes all the work performed

Transaction Management (1)

Business Transaction

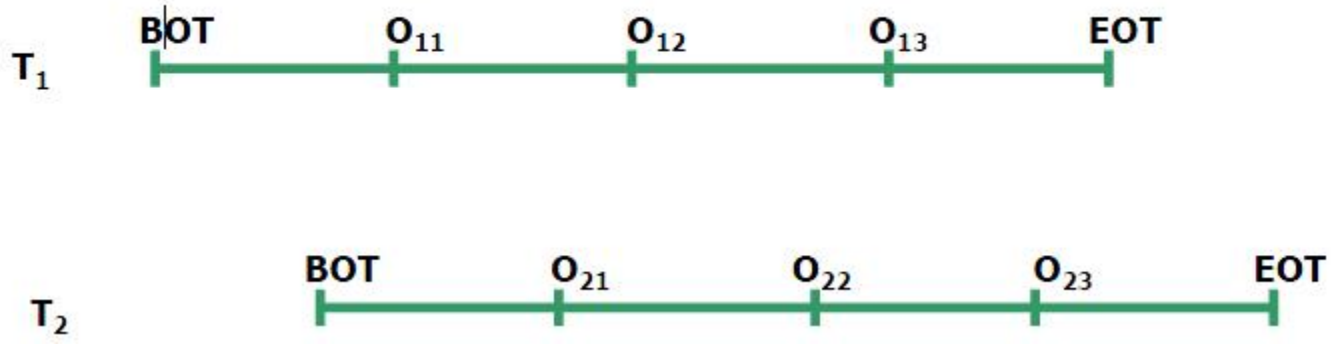
- An interaction in the real world, usually between an enterprise and a person, where something is exchanged.

(On-line) Transaction

- The execution of a program that performs some functions of a business transaction by accessing a shared database, usually on behalf of an online user. (P. Bernstein, E. Newcomer: Transaction Processing, 1997)

Transaction Management

A **transaction** is a collection of operations that performs a single logical function in a database application.



Several operations on the database form a single unit of work.

Transaction-management component ensures that the database remains in a consistent (correct) state despite system failures (e.g. power failures and operating system crashes) and transaction failures.

Concurrency-control manager controls the interaction among the concurrent transactions, to ensure the consistency of the database.

ACID properties

Data integrity means that the data in the database is accurate, complete and consistent both at its creation and at all times during use. Some Data integrity in the database are

- Domains, Unique constraints and primary keys, Foreign keys, Check constraints in table definitions, Check options in view definitions, Default values etc .

For ensuring integrity of the data, the database system need to maintain ACID properties of transaction.

Atomicity:

- Either all operations of the transaction are performed completely or none.

Consistency:

- Execution of a transaction in isolation(that is, with no other transaction executing concurrently) preserves the consistency of the database.
- This requires that the transaction be a correct program.

Isolation:

- Even though multiple transactions execute concurrently, the system guarantees that, for every pair of transactions Ti and Tj, it appears to Ti that either Tj finished execution before Ti started or Tj started execution after Ti finished.
- Each transaction is unaware of other transactions executing concurrently in the system.

Durability:

- After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

Transaction Processing

Transaction Program „Debit-Credit“:

Read message(acctno, tellerno, branchno, delta) from terminal;
BEGIN TRANSACTION
UPDATE ACCOUNT
SET balance TO balance + delta
WHERE acct_no = acctno and balance >= delta
UPDATE TELLER
SET balance TO balance +delta
WHERE teller_no = tellerno
UPDATE BRANCH
SET balance TO balance +delta
WHERE branch_no = branchno
INSERT INTO HISTORY (timestamp, values)
COMMIT TRANSACTION ;
Write message(acctno, balance, . . .) to terminal

Database Users (1)

Naive users:

- are unsophisticated users
- interact with the system by invoking one of the permanent application programs that have been written previously
  - E.g. people accessing database over the web, bank tellers, clerical staff

Application Programmers:

- are computer professionals
- Interacts with the system through DML calls, embedded in a program written in a host language (e.g. C, PL/1, Pascal).
- Uses Rapid Application Development(RAD) tools

Database Users (2)

Sophosticated users

- Interacts with the system without writing programs
- form requests in a database query language
- These are submitted to a query processor that breaks a DML statement down into instructions for the storage manager module
- includes business analysts, scientists, engineers, others thoroughly familiar with the system capabilities. Many use tools in the form of software packages that work closely with the database.

Specialized Users:

- are sophisticated users writing special database application programs. These may
- be CAD systems, knowledge-based and expert systems, complex data systems (audio/video), etc.

Database Administrator – DBA (1)

Is a person having central control of both the data and the program that access those data.

Database administrators duties includes

- Scheme definition:
  - The DBA creates the original database schema by executing set of data definition statements
- Storage structure and access method definition:
  - Writes a set of definitions translated by the data storage and definition language compiler
- Schema and physical organization modification:
  - writing a set of definitions used by the DDL compiler to generate modifications to appropriate internal system tables (e.g. data dictionary).
  - This is done rarely, but sometimes the database schema or physical organization must be modified.

Database Administrator-DBA (2)

- Granting of authorization for data access:
  - granting different types of authorization for data access to various users
- Routine Maintenance
  - Periodically backing up of the database
  - Ensuring enough free space and upgrading disk space
  - Monitoring job running in the system and ensuring performance is not degraded by very expensive task