Introduction to the course

On Line Transaction Processing (OLTP)
- Traditional DBMS usage
- Characterized by:
  - snapshot of current data values
  - detailed data, relational representation
  - structured, repetitive operations
  - read/write access to few records
  - short transactions
  - isolation, reliability, and integrity are critical (ACID)
  - database size \( \approx 100\text{MB-GB} \)

On Line Analytical Processing (OLAP)
- Decision support applications
- Characterized by:
  - "historical" data
  - consolidated, integrated data
  - ad hoc applications
  - read access to millions of records
  - complex queries
  - consistency before and after periodical loads
  - database size \( \approx 100\text{GB-TB} \)

Course content
- First part (weeks 1-7)
  - DBMS server technology
  - SQL Triggers
  - Distributed databases
- Second part (weeks 8-14)
  - Data warehouse design
  - OLAP analysis
  - Data mining

Books
- Course books
  - Tan, Steinbach, Kumar, *Introduction to data mining*, Pearson, 2006

Books
- Other books
  - Kimball e altri, several books and white papers on data warehouse design methodologies and case studies, Wiley
  - Han, Kamber, *Data mining: concepts and techniques*, Morgan Kaufmann, 2006
Database Management Systems

Introduction to DBMS

- Data Base Management System (DBMS)
  - A software package designed to store and manage databases
  - We are interested in internal mechanisms of a DBMS providing services to applications
    - Useful for making the right design choices
      - System configuration
      - Physical design of applications
    - Some services are becoming available also in operating systems

DBMS Architecture

- SQL COMMANDS
  - OPTIMIZER
  - MANAGEMENT OF ACCESS METHODS
  - BUFFER MANAGER
  - CONCURRENCY CONTROL
  - RELIABILITY MANAGEMENT
  - Retrieve block of Data
  - DATABASE

- DBMS SERVER
  - OPTIMIZER
  - MANAGEMENT OF ACCESS METHODS
  - BUFFER MANAGER
  - CONCURRENCY CONTROL
  - RELIABILITY MANAGEMENT
  - Retrieve block of Data
  - DATABASE

DBMS Components

- Optimizer
  - It selects the appropriate execution strategy for accessing data to answer queries
  - It receives in input a SQL instruction (DML)
  - It executes lexical, syntactic, and semantic parsing and detects (some) errors
  - It transforms the query in an internal representation (based on relational algebra)
  - It selects the "right" strategy for accessing data
  - This component guarantees the data independence property in the relational model

- Access Method Manager
  - It performs physical access to data
  - It implements the strategy selected by the optimizer
Database Management Systems  Introduction to DBMS

DBMS Components

- **Buffer Manager**
  - It manages page transfer from disk to main memory and vice versa
  - It manages the main memory portion that is pre-allocated to the DBMS
  - e.g., Oracle SGA

- The memory block pre-allocated to the DBMS is *shared* among many applications

DBMS Components

- **Concurrency Control**
  - It manages concurrent access to data
  - Important for write operations
  - It guarantees that applications do not interfere with each other, thus yielding consistency problems

DBMS Components

- **Reliability Manager**
  - It guarantees correctness of the database content when the system crashes
  - It guarantees atomic execution of a transaction (sequence of operations)
  - It exploits auxiliary structures (log files) to recover the correct database state after a failure

Transaction

- **Transaction** is a logical unit of work performed by an application
  - It is a sequence of one or more SQL instructions, performing read and write operations on the database

- It is characterized by
  - Correctness
  - Reliability
  - Isolation

Transaction example: Bank Transfer

- The following transaction moves 100 euro from account xxx to account yyy

```
UPDATE ACCOUNTS
SET Balance = Balance - 100
WHERE Account_Number = xxx

UPDATE ACCOUNTS
SET Balance = Balance + 100
WHERE Account_Number = yyy
```

Transaction delimiters

- **Transaction start**
  - Typically implicit
  - First SQL instruction
    - At the beginning of a program
    - After the end of the former transaction

- **Transaction end**
  - COMMIT: correct end of a transaction
  - ROLLBACK: end with error
  - The database state goes back to the state at the beginning of the transaction
Transaction end

- 99.9% of transactions commit
- Remaining transactions rollback
  - Rollback is required by the transaction (suicide)
  - Rollback is required by the system (murder)

Transaction properties

- ACID properties of transactions
  - Atomicity
  - Consistency
  - Isolation
  - Durability

Atomicity

- A transaction cannot be divided in smaller units
  - It is not possible to leave the database in an intermediate state of execution
- Guaranteed by
  - Undo. The system undoes all the work performed by the transaction up to the current point
    - It is used for rollback
  - Redo. The system redoes all work performed by committed transactions
    - It is used to guarantee transaction commit in presence of failure

Consistency

- A transaction execution should not violate integrity constraints on a database
- Enforced by defining integrity constraints in the database schema (Create table, ...)
  - Primary key
  - Referential Integrity (Foreign key)
  - Domain Constraints
  - ...
  - When a violation is detected, the system may
    - Rollback the transaction
    - Automatically correct the violation

Isolation

- The execution of a transaction is independent of the concurrent execution of other transactions
- Enforced by the Concurrency Control block of the DBMS

Durability

- The effect of a committed transaction is not lost in presence of failures
  - It guarantees the reliability of the DBMS
  - Enforced by the Reliability Manager block of the DBMS