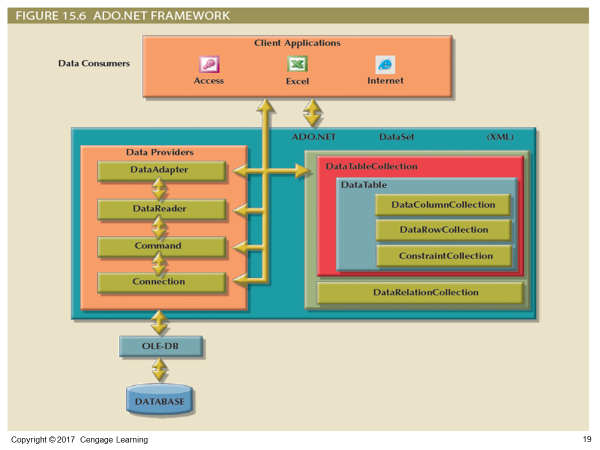
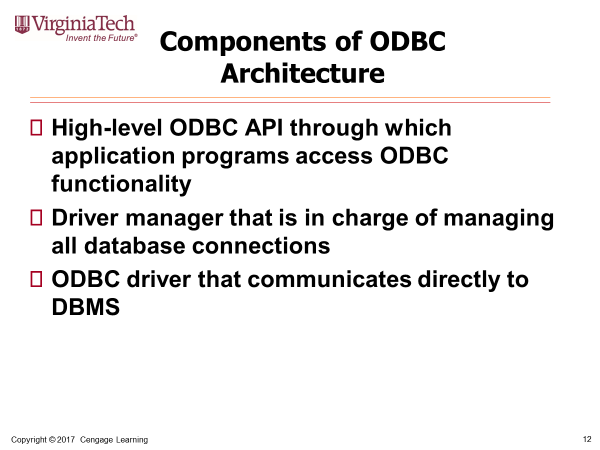
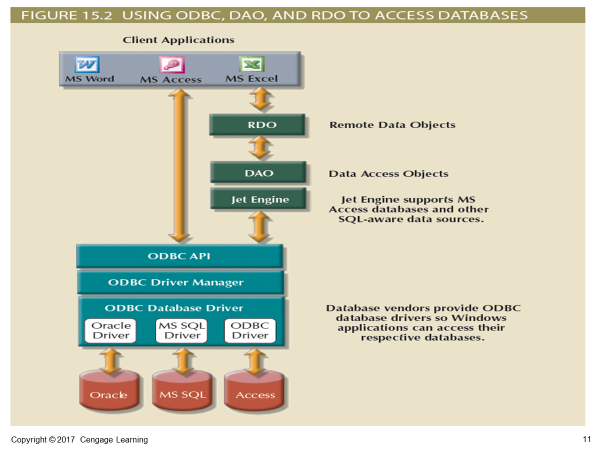
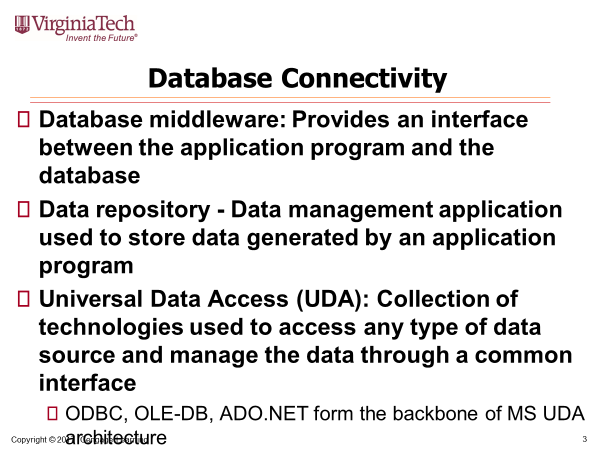
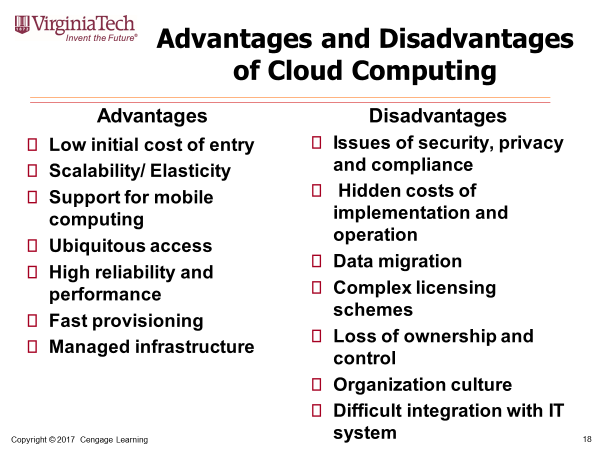
* **Remote Data Objects (RDO)**
* **Dynamic-link libraries (DLLs)**
* **Data Access Objects (DAO):** Object-oriented API used to access MS Access, MS FoxPro, and dBase databases from Visual Basic programs
* **Open Database Connectivity (ODBC):**Microsoft’s implementation of a superset of SQL Access Group Call Level Interface (CLI) standard for database access
* **Object Linking and Embedding for Database (OLE-DB):** Database middleware that adds object-oriented functionality for access to data









**Transaction Properties (ACIDS)**

* **Atomicity**
  + All operations of a transaction must be completed
* **Consistency** 
  + Permanence of database’s consistent state
* **Isolation** 
  + Data used during transaction cannot be used by second transaction until the first is completed
* **Durability** 
  + Once transactions are committed, they cannot be undone
* **Serializability**
  + Concurrent execution of several transactions yields consistent results
* **Write-ahead log protocol: Ensures that transaction logs are always written before the data are updated**
* **Redundant transaction logs: Ensure that a physical disk failure will not impair the DBMS’s ability to recover data**
* **Buffers:** **Temporary storage areas in a primary memory**
* **Checkpoints:** **Allows DBMS to write all its updated buffers in memory to disk**
* **Deferred-write technique** 
  + Only transaction log is updated
* **Write-through technique**
  + Database is immediately updated by transaction operations during transaction’s execution

**Concurrency Control Problems**

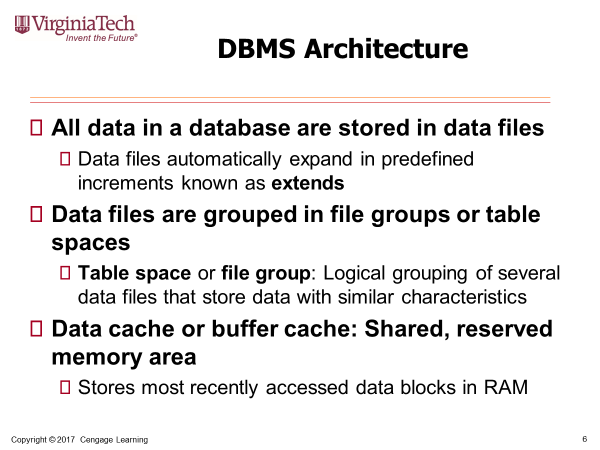
* Lost update
  + Occurs in two concurrent transactions when:
    - Same data element is updated
    - One of the updates is lost
* Uncommitted data
  + Occurs when:
    - Two transactions are executed concurrently
    - First transaction is rolled back after the second transaction has already accessed uncommitted data
* Inconsistent retrievals
  + Occurs when a transaction accesses data before and after one or more other transactions finish working with such data
* **Three deadlock control techniques:** prevention, detection, and avoidance
* **Optimistic methods assume the majority of database transactions do not conflict**
  + Transactions are executed concurrently, using private copies of the data

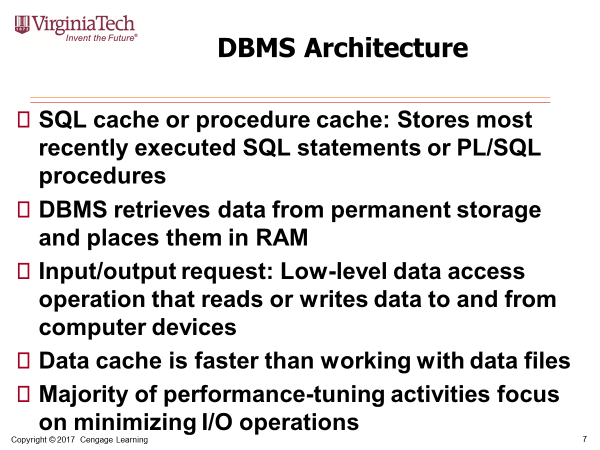
**Two-Phase Locking (2PL)**

* **Phases**
  + Growing phase - Transaction acquires all required locks without unlocking any data
  + Shrinking phase - Transaction releases all locks and cannot obtain any new lock

**Database performance tuning:** Set of activities and procedures that reduce response time of database system

SQL performance tuning: on the client side



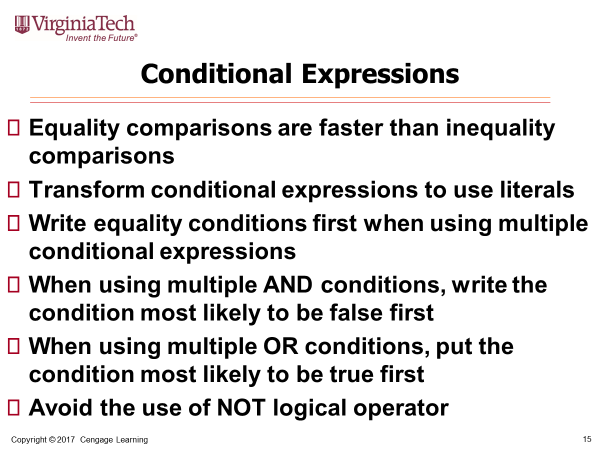


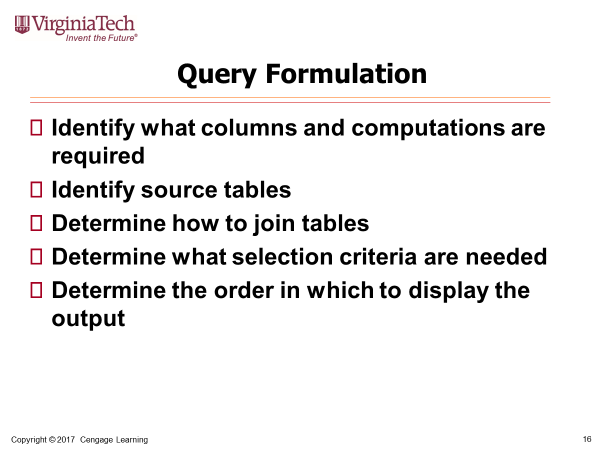
* **Automatic query optimization: DBMS finds the most cost-effective access path without user intervention**
* **Manual query optimization: Requires that the optimization be selected and scheduled by the end user or programmer**
* **Static query optimization: best optimization strategy is selected when the query is compiled by the DBMS**
  + Takes place at compilation time
* **Dynamic query optimization: Access strategy is dynamically determined by the DBMS at run time, using the most up-to-date information about the database**
  + Takes place at execution time
* **Statistically based query optimization algorithm: Statistics are used by the DBMS to determine the best access strategy**
* **Statistical information is generated by DBMS through:**
  + **Dynamic statistical generation mode**
  + **Manual statistical generation mode**
* **Rule-based query optimization algorithm: based on a set of user-defined rules to determine the best query access strategy**
* **Query Processing Steps:** 
  + **Parsing:** 
    - **Access plans are DBMS specific**
    - **Translate client’s SQL query into a series of complex I/O operations**
    - **Required to read the data from the physical data files and generate result set**
  + **Execution:**
    - **All I/O operations indicated in the access plan are executed**
    - Locks are acquired
    - Data are retrieved and placed in data cache
    - Transaction management commands are processed
  + **Fetching:**
    - **Rows of resulting query result set are returned to client**
    - DBMS may use temporary table space to store temporary data
    - Database server coordinates the movement of the result set rows from the server cache to the client cache
* Fully equivalent
  + Optimized query results are always the same as original query
* More efficient
  + Optimized query will almost always execute faster than original query
* **Delay introduced in the processing of an I/O operation that slows the system**
* **Caused by the:**
  + CPU
  + RAM
  + Hard disk
  + Network
  + Application code

**Data sparsity:** number of different values a column could possibly have

**Indexes implemented using:**

* + **Hash indexes**
  + **B-tree indexes (most common)**
  + **Bitmap indexes**
* **Rule-based optimizer (based on rules)**
  + Preset rules and points
  + Rules assign a fixed cost to each operation
* **Cost-based optimizer (based on data)**
  + Algorithms based on statistics about objects being accessed
  + Adds up processing cost, I/O costs, resource costs to derive total cost
* **Optimizer hints:** Special instructions for the optimizer, embedded in the SQL command text
* **Majority of performance problems are related to poorly written SQL code**
* **Indexes are used when:**
  + Indexed column appears by itself in search criteria of WHERE or HAVING clause
  + Indexed column appears by itself in GROUP BY or ORDER BY clause
  + MAX or MIN function is applied to indexed column
  + Data sparsity on indexed column is high
  + When a subset of rows from a large table is to be selected based on a given condition
  + Indexing is a problem for insert statements
* **General guidelines for indexes:**
  + Create indexes for each attribute in WHERE, HAVING, ORDER BY, or GROUP BY clause
  + Do not use in small tables or tables with low sparsity
  + Declare primary and foreign keys so optimizer can use indexes in join operations
  + Declare indexes in join columns other than PK/FK
* **Guidelines to write efficient conditional expressions in SQL code**
  + Use simple columns or literals as operands
  + Numeric field comparisons are faster than character, date, and NULL comparisons





* During query optimization, DBMS chooses:
  + Indexes to use, how to perform join operations, table to use first, etc
* **In-memory database:** Store large portions of the database in primary storage
* **Recommendations for physical storage of databases:**
  + Use **RAID** (Redundant Array of Independent Disks) to provide a balance between performance improvement and fault tolerance
  + Minimize disk contention
  + Put high-usage tables in their own table spaces
  + Assign separate data files in separate storage volumes for indexes, system, and high-usage tables
* **Index-organized table** or **clustered index table**: Stores the end-user data and the index data in consecutive locations in permanent storage
* **Factors That Aided DDBMS to Cope With Technological Advancement** 
  + Acceptance of Internet as a platform for business
  + Mobile wireless revolution
  + Usage of application as a service
  + Focus on mobile business intelligence

**Desirability of Distributed DBMS Over Centralized DBMS**

* Performance degradation
* High costs
* Reliability problems
* Scalability problems
* Organizational rigidity

**Characteristics of Distributed Management Systems**

* **Application interface**
* **Validation**
* **Transformation**
* **Query optimization**
* **Mapping**
* **I/O interface**
* **Formatting**
* **Security**
* **Backup and recovery**
* **DB administration**
* **Concurrency control**
* **Transaction management**

**Transaction processor (TP):** Software component of a system that requests data

* + Also known as **transaction manager (TM)** or **application processor (AP)**

**Data processor (DP) or data manager (DM)**

* + Software component on a system that stores and retrieves data from its location

Homogeneous, **Heterogeneous DDBMSs, Fully heterogeneous DDBMSs**

**Replica transparency**: DDBMS’s ability to hide multiple copies of data from the user

* **Network latency**: delay imposed by the amount of time required for a data packet to make a round trip
* **Network partitioning**: delay imposed when nodes become suddenly unavailable due to a network failure
* **Data fragmentation, data replication**
* **Horizontal fragmentation: Division of a relation into subsets (fragments) of tuples (rows)**
* **Vertical fragmentation: Division of a relation into attribute (column) subsets**
* **Mixed fragmentation: Combination of horizontal and vertical strategies**
* **Data Replication:** Data copies stored at multiple sites served by a computer network, Helps restore lost data
* **Mutual consistency rule:** Replicated data fragments should be identical
* **Styles of replication**
  + Pushreplication
  + Pullreplication
* **CAP stands for:**
  + Consistency
  + Availability
  + Partition tolerance
* **Basically available, soft state, eventually consistent (BASE)**
* **Distributed Database Transparency Features**
* Distribution transparency
* Transaction transparency
* Failure transparency
* Performance transparency
* Heterogeneity transparency
* **Distribution Transparency:** allows management of physically dispersed database as if centralized
* **Levels**
  + **Fragmentation transparency**
  + **Location transparency**
  + **Local mapping transparency**
* **Unique fragment:** Each row is unique, regardless of the fragment in which it is located
* **Supported by distributed data dictionary (DDD) or distributed data catalog (DDC)**
  + DDC contains the description of the entire database as seen by the database administrator
* **Distributed global schema:** Common database schema to translate user requests into subqueries
* **Dirty data**
  + Data that suffer from inaccuracies and inconsistencies
  + Threat to organizations
* **Data quality**
  + Comprehensive approach to ensuring the accuracy, validity, and timeliness of the data
* **Data profiling software** 
  + Consists of programs that gather statistics and analyze existing data sources
* **Master data management (MDM) software**
  + Helps prevent dirty data by coordinating common data across multiple systems
* Technological aspect
  + Selecting, installing, configuring, and monitoring the DBMS to ensure that it operates efficiently
* Managerial aspect
  + Careful planning to create an appropriate organizational structure
* Cultural aspect
  + Listening to people’s concerns about the system and explaining its uses and benefits
* **Information systems (IS) department**
  + Provides end users with data management support and solutions for information needs
* **Database administrator**
  + Responsible for control of the centralized and shared database
* **Systems administrator**
  + General coordinator of all DBAs
* **Data administrator (DA)** or **information resource manager (IRM)**
  + Has a higher degree of responsibility and authority than the DBA

**DBA’s Managerial Role**

* **Provide end-user support**
* **Enforce policies, procedures, and standards for correct data creation, usage, and distribution within the database**
* **Manage data security, privacy, and integrity**
* **Manage data distribution and use**
* **Manage data backup and recovery**
  + Fully recover data in case of data loss
  + **Database security officer (DSO)**:Ensures database security and integrity
  + **Disaster management**: Planning, organizing, and testing of database contingency plans and recovery procedures
  + **Full backup** or **database dump**: Produces a complete copy of the entire database
  + **Incremental backup**: Produces a backup of all data since the last backup date
  + **Concurrent backup**: Takes place while the user is working on the database

**DBA’s Technical Role**

* **Evaluate, select, and install DBMS and related utilities**
* **Design and implement databases and applications**
* **Test and evaluate databases and applications**
* **Operate the DBMS, utilities, and applications**
* **Train and support users**
* **Maintain the DBMS, utilities, and applications**
* **Confidentiality:** Protecting data against unauthorized access
* **Compliance:** Activities that meet data privacy and security reporting guidelines
* **Integrity:** Keeping data consistent and free of errors or anomalies
* **Availability:** Accessibility of data whenever required by authorized users and for authorized purposes
* **Security Vulnerability:** Weakness in a system component that could allow unauthorized access or cause service disruptions
  + **Categories -** Technical, managerial, cultural, and procedural
  + **Security threat:** Imminent security violation
  + **Security breach:** Occurs when a security threat is exploited and could lead to a database whose integrity is preserved or corrupted
* **Authorization management:** Procedures to protect database security and integrity
  + User access management
  + View definition
  + DBMS access control
  + DBMS usage monitoring
    - **Audit log**: Automatically records description of database operations performed by all users

**Data Dictionary**

* **Types** 
  + Integrated - Included with the DBMS
  + Standalone - Third-party systems
* **Active data dictionary:** Automatically updated by the DBMS with every database access
* **Passive data dictionary:** Requires running a batch process
* **Main function -** Store description of all objects that interact with the database
* **Metadata is the basis for monitoring database use and for assigning access rights to users**
* **DBA uses data dictionary to support data analysis and design**
* **Governance:** Methods for controlling business health and for consistent decision making
* **Key performance indicators (KPI):** Numeric or scale-based measurements that assess company’s effectiveness in reaching its goals
* **Datawarehouse Components:**
  + Integrated
  + Subject-Oriented
  + Time-Variant
  + Non-volatile