**INFO90002 Database Systems & Information Modeling**

**期末考试机经(精修版)**

**Week 7: Database & Application Software**

**Outline:**

* Reasons we put applications between users and databases (1)
* Presentation / UI
* Business logic
* Embedding SQL in application software (2, 3)
* 1, 2, 3-tiered architectures (4, 5)
* Web applications (6, 7)

1. **Why do we put applications between users and databases?**

* Presentation logic: How business objects are displayed to users of the software
* Input (keyboard, touchscreen, voice, etc.)
* Output (large screen, printer, phone, ATM, etc.)
* Business logic: The part of the program that encodes the real-world business rules that determine how data can be created, stored, and changed.
* Input and command handling
* Enforcement of business rules
* Storage logic:
* Persistent storage of data
* Enforcement of data integrity

1. **How does SQL combine data manipulation with procedural language to handle sequence, iteration, decision?**

* Embedded SQL:
* SQL embedded in code is interpreted and replaced with library calls
* Dynamic SQL:
  + Host language sends SQL to DBMS via middleware e.g. ODBC
  + Data is passed back to program as record-set
  + Host language can handle business and presentation logic
* Stored Procedure & Trigger:
  + Procedural code is stored and executed in the DBMS
  + Enforce business logic within the database

1. **What are advantages and disadvantages of stored procedures and triggers?**

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| * Faster code execution * Improved security and data integrity * Business logic under control of DBA | * Code is not under the control of the application programmer * Proprietary language |

1. **Draw a diagram of a 1/2/3/4-tier application architecture. Label each component.**

* 1-tier architecture: Entire application ran on the same computer (Embedded SQL)
* 2-tier architecture: clients (presentation, business logic) <--> DBMS (storage logic)
* 3-tier architecture: clients (presentation logic) <--> web server (business logic) <--> DB server (storage logic)
* Web-based 3-tier architecture:
* Browser handles presentation logic
* Browser talks to web server via simple, standard protocol
* Business logic and data storage handled on server(s)
* 4-tier architecture: clients <--> web server <--> app server <--> DB server

1. **What are the advantages and disadvantages of 2/3 tier application architectures?**

|  |  |  |
| --- | --- | --- |
|  | **Advantages** | **Disadvantages** |
| **2-Tier** | * Clients and servers share processing load * Good data integrity since data is all processed centrally * Stored procedures allow some business rules to be implemented on the database server | * Presentation, data model, business logic are intertwined at client * If DB schema changes, all clients break * DB connection for every client, thus difficult to scale |
| **3-Tier** | * Scalability: more scalable than 2-tier * Long-term cost reduction * Improved security: customer machine does presentation only | * High short-term costs * More Tools and training * Complex to design |
| **Web**  **Based**  **3-Tier** | * Everyone has a browser * HTML and HTTP are simple standards, widely supported * Opens up the possibility of global access to database | * Even more complexity in the middle-tier * Simple standards = hard to make complex application * Global access = potential security nightmare |

1. **How has the web impacted database design and administration?**

* Browser handles presentation logic
* Browser talks to web server via simple, standard protocol
* Business logic and data storage handled on server(s)

1. **Why do we create web applications?**

* Web browsers are ubiquitous
* No need to install client software for external customers
* Simple communication protocols

**Week 8-1: Transactions and Concurrency**

**Outline:**

* Definition of a transaction
* Definition (1)
* Purposes (2)
* Transaction properties
* ACID (3) & Serializability (4)
* Transaction logging (5)
* Concurrency
* Problems:
* Lost updates, uncommitted data, inconsistent retrieval (6)
* How to handle these – the scheduler (7)
* Locking (8) and its various types (9) and granularity (10)

1. **What does transaction mean?**

Transaction means a logical unit of work that must either be entirely completed or aborted (indivisible, atomic).

1. **Explain two different purposes for using transactions.**

* Users need the ability to define a unit of work.
* Concurrent access to data by more than one user or program.

1. **Describe the four properties of transactions referred to by the acronym “ACID”.**

* Atomicity: A transaction is treated as a single, indivisible, logical unit of work. All operations in a transaction must be completed; if not, then the transaction is aborted. (要么一起活，要么谁也别想活)
* Consistency: Constraints that hold before a transaction must also hold after it (multiple users accessing the same data see the same value).
* Isolation: Changes made during execution of a transaction cannot be seen by other transactions until this one is completed. (David课上悄悄咪咪说了一句，如果isolation不存在的话，会造成uncommitted data，hint! Hint!)
* Durability: When a transaction is complete, the changes made to the database are permanent, even if the system fails.

1. **What is serializability of a transaction?**

* Transactions ideally are “serializable”
* Multiple, concurrent transactions appear as if they were executed one after another
* Ensures that the concurrent execution of several transactions yields consistent results
* But true serial execution (i.e no concurrency) is very expensive!

1. **What is transaction logging?**

* To restore database to a previous consistent state, DBMS tracks all updates to data
* This transaction log contains:
* A record for the beginning of the transaction
* For each SQL statement
* The ending (COMMIT) of the transaction
* Also provides the ability to restore a corrupted database.

1. **Explain the three concurrency problems (Lost updates他考了两次叫你画timeline然后问加了锁后怎么画，今年他要是骚起来可能会考另外两个加了锁后怎么画。)**

* Lost updates: Occurs when two concurrent transactions are updating the same data element and one of the updates is lost (overwritten by the other transaction).
* Uncommitted data: Occurs when two transactions execute concurrently and the first is rolled back after the second has already accessed the uncommitted data.
* Inconsistent retrievals: Occurs when one transaction calculates some aggregate functions over a set of data, while other transactions are updating the data.

1. **What are the three concurrency control methods?**

* Locking: the main method used
* Timestamp:
* Assigns a global unique timestamp to each transaction
* Each data item accessed by the transaction gets the timestamp
* Thus for every data item, the DBMS knows which transaction performed the last read or write on it
* When a transaction wants to read or write, the DBMS compares its timestamp with the timestamps already attached to the item and decides whether to allow access
* Optimistic:
* Based on the assumption that the majority of database operations do not conflict
* Transaction is executed without restrictions or checking
* Then when it is ready to commit, the DBMS checks whether it any of the data it read has been altered – if so, rollback.

1. **What is the purpose of locking?**

* Guarantee exclusive use of a data item to a current transaction.
* Required to prevent another transaction from reading inconsistent data.

1. **What are the types of lock?**

* Binary lock:
* Has only two states: locked (1) and unlocked (0).
* Eliminate “Lost Update” problem: The lock is not released until the statement is completed
* Considered too restrictive to yield optional concurrency, as it locks even for two READs (when no update is being done)
* Exclusive lock:
* Access is reserved for the transaction that locked the object.
* Must be used when transaction intends to WRITE
* Granted if and only if no other locks are held on the data item
* Shared lock:
* Other transactions are also granted Read access
* Issued when a transaction wants to READ data, and no Exclusive lock is held on that data item: Multiple transactions can each have a shared lock on the same data item if they are all just reading it.

1. **Locking can occur at different levels – explain.**

* Database-level lock
* Table-level lock
* Page-level lock
* Row-level lock
* Field-level lock

**Week 8-2: Distributed Databases**

**Outline:**

* Define the term distributed database (1)
* Advantages and disadvantages (2)
* Looks at the objectives and trade-offs in distributed database
* Ease of access / location transparency, local autonomy (3)
* Asynchronous / synchronous updates (4)
* Data replication
* Types (5)
* Advantages and disadvantages (6)
* Description and comparison of options (我觉得不会考)

1. **What is a distributed database and decentralized database?**

* Distributed database:
* A single logical database physically spread across multiple computers in multiple locations that are connected by a data communications link.
* Appears to users as though it is one database.
* Decentralized database
* A collection of independent databases which are not networked together as one logical database.
* Appears to users as though many databases.

1. **What are the advantages and disadvantages of a distributed database?**

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| * Good fit for geographically distributed organizations/users * Faster data access (to local data) * Faster data processing | * Complexity of management and control * Data integrity: additional exposure to improper updating * Security: many server sites means higher change of breach |

1. **What are the objectives and tradeoffs of a distributed database?**

* Objectives:
  + Location transparency: a user does not need to know where particular data are stored.
  + Local autonomy: a node can continue to function for local users if connectivity to the network is lost
* Tradeoffs:
  + Synchronous
  + Asynchronous

1. **Describe the advantages and disadvantages of synchronous and asynchronous updates.**

|  |  |  |
| --- | --- | --- |
|  | **Advantages** | **Disadvantages** |
| **Synchronous** | * Data is continuously kept up to date * If any copy of a data item is updated anywhere on the network, the same update is immediately applied to all other copies or it is aborted. * Ensure data integrity and minimizes the complexity of knowing where the most recent copy of data is located. | * Can result in slow response time and high network usage |
| **Asynchronous** | * Acceptable response time * Suits some information systems more than others | * Some delay in propagating data updates to remote databases * May be more complex to plan and design |

1. **Describe the differences between replication and partitioning.**

* Data replication: Data copied across sites
* Horizontal partitioning: Table rows distributed across sites
* Vertical partitioning: Table columns distributed across sites

1. **Discuss the advantages and disadvantages of different distributed database options.**

|  |  |  |
| --- | --- | --- |
|  | **Advantages** | **Disadvantages** |
| **Replication** | * High reliability due to redundant copies of data * Fast access to data at the location where it is most accessed * May avoid complicated distributed integrity routines | * Need more storage space * Can retrieve incorrect data if updates have not arrived * Takes time for update operations |
| **Horizontal Partitioning** | * Efficiency: Data stored close to where it is used * Better performance: Local access optimization * Security: Only relevant data is stored locally | * Inconsistent access speed: Accessing data across partitions * Backup vulnerability: No data replication |
| **Vertical**  **Partitioning** | Same as above | Same as above   * Combining data across partitions is more difficult because it requires joins (instead of unions) |

**Week 9: Database Administration**

**Outline:**

* The DBA and Data Administration roles (1)
* Architecture of a RDBMS
  + Components (2, 3) and their interactions (4)
* Performance
  + What effects database performance (5)
  + Common approaches to monitoring and tuning performance
* Security
  + Threats (8)
  + Access control (9) and encryption (10)
  + Web app security, SQL injection (11, 12)
* Backup and recovery
  + Types of failure (13), types of backups (14)
  + Other methods to reduce likelihood of data loss (15)

1. **Describe and differentiate the DBA and DA roles. (真不知道这tm有什么好考的！)**

|  |  |
| --- | --- |
| **Data Administrator (management role)** | **Database Administrator (technical role)** |
| * Set data policies, procedures and standards * Planning * Resolve data conflicts | * Analyze and design DB * Select DBMS / tools / vendor * Install and upgrade DBMS |

1. **What are the components of DBMS? (这张表简写一下，屁屁踢上写的没一句人话鬼知道他在讲啥)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Query Processing** | | | |
| **Parser / Compiler** | **Optimizer** | | **Executor** |
| Check if the syntax is correct and if the user has permission. | Choose the best options of doing things with the lowest cost. (就是执行一个指令有几条路可以选，系统会选最好走的那条) | | Meet the ACID test |
| **Concurrency Control** | | | |
| **Transaction Manager** | | **Lock Manager** | |
| Handles all aspects of the SQL transaction – which DBMS user wants WHAT resource | | A list of what resources are locked and by which user at what level | |
| **Storage** | | | |
| **File & Access Methods** | **Buffer Pool** | | **Disk Space Management** |
| * How to get stuff from disk to memory * How to read from a file | The place where all data live | | * How to organize growth of data on disk efficiently by writing efficiently. (可以想象成一个仓库管理员，差不多就这个意思) |
| **Crash Recovery** | | | |
| **Log Manager** | | | |
| * Records ALL changes: Statement, Transaction, Database | | | |

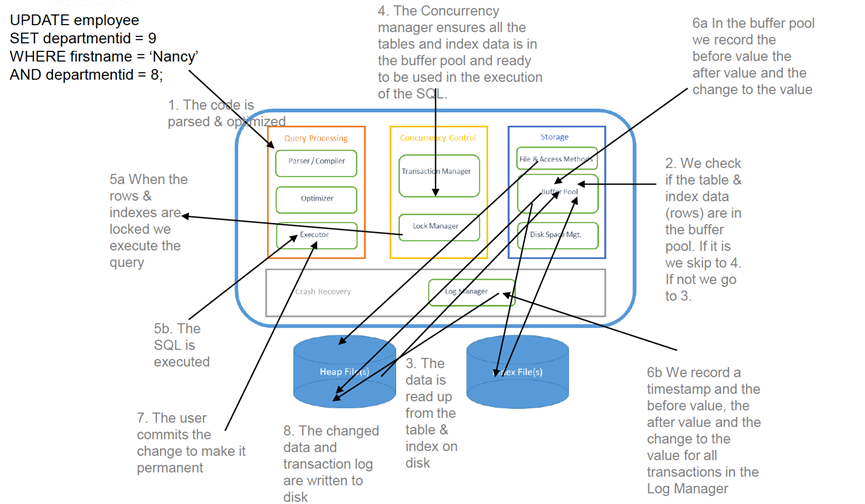
1. **How does the Buffer Pool work, and what is its purpose?**

* Working process:

Read many rows into a buffer, which is the smallest unit of work, and store the blocks, which constituted by buffers, into the disks.

* Purpose:
  + Reading and writing from disks are expensive and slow, memory could efficiently avoid some reads or writes, and the goal is to minimize reads and writes.
  + To handle the concurrency of operations which is expensive, so it is better to put as much as them into memory than into disks**.**

1. **What is the interaction of different components in DBMS?**



1. **What affects database performance?**

* Caching data in memory, e.g. data buffers
* Placement of data files across disc drives
* Use of fast storage such as SSD

1. **What are the common approaches to improve database performance?**

* Caching data in memory: Goal is to minimize reads
* Data file location & fast disks (SSD): Spread the files across the physical server
* Distribution & replication

1. **What are the threats to databases?**

* Loss of integrity
  + Keep data consistent
  + Free of errors or anomalies
* Loss of availability
  + Must be available to authorized users for authorized purposes
* Loss of confidentiality
  + Must be protected against unauthorized access

1. **What is access control?**

Access control is handled by the DBA creating user accounts for those with a legitimate need to access the DB

1. **What is encryption?**

* Particular tables or columns may be encrypted to:
  + Protect sensitive data (e.g. password) when they are transmitted over a network
  + Encrypt data in the database (e.g. credit card number)
* Data is encoded using an algorithm
  + Authorized users are given keys to decipher data

1. **How does an “SQL Injection” attack work? (把那个例子一起看一下)**

* SQL injection is a technique used to exploit web applications that use user input within database queries.
* Malicious code is entered into a data entry field in such a way that it becomes part of SQL commands that are run against the database.

1. **How can we protect against “SQL Injection”?**

* Primary defenses:
  + Prepared statement (parameterized queries)
  + Stored procedures
  + i.e. “escape” all user input
* Additional defenses:
  + Principle of least privilege
  + White list input validation

1. **What are the categories of failures?**

* Statement failure
* User process failure
* Network failure
* User error
* Instance failure
* Media failure

1. **What are the different types of backup?**

|  |  |
| --- | --- |
| **Physical backup** | **Logical backup** |
| * Raw copies of files and directories * Suitable for large databases that need fast recovery * Backup should include logs | * Backup completed through SQL queries * Slower than physical * Doesn’t include log or config files |
| **Online (or HOT) backup** | **Offline (or COLD) backup** |
| * Backup occur when the database is “live” * Clients don’t realize a backup is in progress * Need to have appropriate locking to ensure integrity of data | * Backups occur when the database is stopped * To maximize availability to users, take backup from replication server not live server * Simpler to perform |
| **Full backup** | **Incremental backup** |
| * A full backup is where the complete database is backed up * It includes everything you need to get the database operational in the event of a failure | * Only the change since last backup are backed up * For most databases this means only backup log files |
| **Onsite backup** | **Offsite backup** |
| * All databases are set in one place. | * Enables disaster recovery (because backup is not physically near the disaster site) |

1. **Describe other methods to reduce likelihood of data loss.**

* Server replication
* Server cluster
* RAID attributes

**Week 10 and 12: NoSQL and Industry Trends**

**Outline:**

* The relational status-quo
  + Major vendors (1)
  + Current offerings (2)
* Challenges to the relational status-quo
  + Big data (2, 3, 4)
  + Cloud storage (5, 6, 7)
  + Object-oriented applications (8)
* Responses
  + NoSQL (9, 10, 11)
  + NewSQL
  + Hadoop
  + In-memory databases

1. **Who are the leading commercial and open-source vendors?**

Oracle, Microsoft SQL Server, IBM DB2

1. **What are the current major relational offerings?**

Oracle Database 18c, Microsoft SQL Server 2017, MySQL 8.0

1. **What is big data?**

* 4V’s of big data:
  + Volume (scale of data)
  + Velocity (analysis of streamlining data)
  + Variety (different form of data)
  + Veracity (uncertainty of data)
* Difficult to process high volume and velocity of data using traditional database management and processing tools

1. **What are the solutions to big data?**

Distributed database

* Vertical scaling (bigger server): gives diminishing returns
* Horizontal scaling: spread data across many small servers
* That inspired NoSQL database

1. **What are the problems of big data?**

* In its infancy: Tools for using and analyzing big data, as well as standards, are still being developed.
* Need the right talent and technology and structure of workflows to optimize the use of big data
* Requires expensive professionals (“data scientists”)

1. **What is Database as a service (DBaaS)?**

* Similar to other cloud services
  + DBMS and data are in the cloud
  + Your application connects as required
  + Pay per usage: quantity data stored, input/output
  + DBMS is administered by cloud provider
    - reduces need for in-house DBA
    - may be managed via web console
* Can be presented as either
  + Database as a service (DBaaS)
* Relational database
* Non-relational (NoSQL) database
  + Virtual machine with a database installed

1. **What are the advantages of cloud storage?**

Simplifies setting up, and especially scaling up, your database

1. **What are the potential cloud database issues?**

* Data security:
  + Provider may not have fully integrated security structure
  + May need to resort to encryption
* Legal framework
  + Need to (continue to) conform to laws governing use of data
* Large movements of data between your site and cloud
  + During initial setup
  + During ongoing integration with local data

1. **What is an object-oriented database?**

* Direct storage of object-oriented data
* Offers benefits of OO and Relational
  + Direct persists objects
  + Offers performance and security of relational DB
* Reason of not popular:
  + Often tied to particular OO programming languages
  + Lack of compatibility between different OODMBSs
  + Lack of standard ad hoc query language

1. **What are the pros and cons of relational databases?**

|  |  |
| --- | --- |
| **Pros** | **Cons** |
| * Flexible, suits any data model * Standard interface language SQL * Ad-hoc queries, across and within “data aggregates” | * Object-Relational impedance mismatch * Not good with big data * Not good with clustered/replicated servers |

1. **What is a NoSQL database?**

* Features
  + Doesn’t use relational model or SQL language
  + Designed to run on distributed servers
  + Most are open-source
  + Built for the modern web
  + Schema-less (though “implicit schema” in the application) (把后面那个MongoDB的例子看一下，去年tm竟然考了一次)
  + ‘eventually consistent’
* Goals
  + To improve programmer productivity (OR mismatch)
  + To handle larger data volumes and throughput (big data)

1. **Describe the four main types of NoSQL database**

* Key-value store:
  + key = primary key
  + Value = anything.
  + The application is in charge of interpreting what it means. Usually used to store photos
* Document databases:
  + Like a key-value db, except that the "value" (document = object represented as JSON file) is "examinable" by the db, so its contents can be queried and updated
* Column families:
  + It contains many ‘rows”, but each row can store a different set of columns.
  + Columns rather than rows are stored together on disk.
  + Makes analysis by column faster – not OLTP
* Key value, document store and column-family are “aggregate-oriented” databases.
  + Pros:
    - Entire aggregate of data is stored together
    - Efficient storage on clusters / distributed databases
  + Cons:
* Hard to analyze across subfields of aggregates
* Graph database:
  + Using a nodes-and-arc network
  + Consists of notes and links between the notes
  + To stores entities and their relationships

1. **What is CAP theorem?**

* CAP Theorem says you can only have two out of three of Consistency, Partition Tolerance, and Availability.
  + Consistency: Everyone always sees the same data
  + Availability: System stays up when nodes fail
  + Partition Tolerance: Systems stays up when network between nodes fail
  + Most NoSQL lives between AP, Oracle RAC lives between AC.
* If you have a distributed database, when a partition occurs, you must then choose consistency or availability.

1. **What is BASE?**

* Basically availability: This constraint states that the system does guarantee the availability of data; there will be a response to any request. But data may be in an inconsistent or changing state.
* Soft state: The state of the system could change over time – even during times without input there may be changes going on due to “eventual consistency”. (即薛定谔的系统，处于一种未知的中间状态，状态会随着系统的心情不停变化)
* Eventual consistency: The system will eventually become consistent once it stops receiving input. The data will propagate to everywhere it needs to, sooner or later, but the system will continue to receive input and is not checking the consistency of every transaction before it moves onto the next one. (一旦停止输入，系统就不皮了，但心情可能依旧鳝变)

1. **What is new SQL?**

* Like NoSQL, but offers ACID transactions
* Suited where consistency is important, e.g. financial data
* Offers traditional relational features like SQL, transactions
* But scales horizontally like NoSQL systems
* Distributed, automatic sharding

1. **What is Hadoop?**

For data-processing, not operational database

1. **What is in-memory database?**

* Data are stored in memory and processed there
* Disk is for long-term persistence
* Primary motive is speed – of storage, transactions, analytics