```
# INFO90002-Database-Systems-and-Information-Modelling
# Author: Chongzheng Zhao(https://github.com/ChongzhengZhao/)
# 2019 Semester 2
#0 Physical Design + Normalization
- When to create index for colomn?
	- in where clause (Query frequently)
	- used for join (Need to search PK in another table)
	- large table
	- colomn which value do not change often
	- colomn which less than 15% access frequently
	- value scope is large (Name)
	- value scope is small (Gender)
	- automatic create: Primary keys, Foreign keys, unique colomns
- Object of Normalization and Denormalization
	-.. .. ?

- What is data integrity?

#1 DB Application / Web Application
- SQL has some flawnesses to deal with business problem
	- Need to handle #sequence, #iteration and #decision
	- we can make bussiness logic within DB, to do:
		- #Embedded SQL
			- SQL statement is embedded in code
		- #Dynamic SQL
			- programming language send SQL statement to DBMS via middleware
			- programming language can deal with business and presentation logic
- System Architecture
	- Presentation logic
		input & output
	- Business logic
		input & command handling & follow commercial rule
	- Storage logic
		persistent storage & follow data integrity rule
- two-tier architechture - Client <=> DB Server
	- Benefits: Both share processing load; Good data integrity
	- Cons: Hard to scale; Three logic are interwined at server; Client break if schema change;
```

- three-tier architechture - Clinet <=> Application Server <=> DB server
	- Benefits: Scalablity; Security; Flexibility; Long-term cost less
	- Cons: Short-term cost more; Hard to design
- [2017]four-tier architechture - Client <=> Web Server <=> Application Server <=> DB server
	- Benefits: Broswer handle presentation logic; Easy to use(Browser is everywhere); Globle access
	- Cons: Cannot make complex application; Security problem; Middle-tier is complex
- [2017]why create web application? How web impact database design?(Benefits of four-tier archit)
	- Broswers are everywhere
	- No need to install DB-client software
	- Simple protocol
	- OS & Web server independent
	- Reduce implementation time & cost

#2 Transaction
- What is transaction?
	- Transaction can be regarded as #atomic, which is a logical unit of work must #either be completed or aborted.
	- a successful transaction change the DB from one #consistent status to another consistent status
- [2016]Function of transaction(Why use transaction?)
	- define a #unit of work
	- multiple user can #concurrency access to data
	- perform #Undo in DB manipulation
- Transaction SQL: Begin(Write Statement), Commit(Approval), Rollback(Undo)
- Transaction Properties: ACID & Serializability
- What is Serializabity? Excute one after another, yield consistent results
I- [2017]What is ACID?
	- Atomicity : Treat transaction as single, invisible, logical unit of work
	- Consistency: Constrains before transaction is hold after transaction done
	- Isolation : Invisible during transaction
	- Durablity : Transaction made permenent change.
I- Concurrency Problem: [Draw a figure!]
	- [2017]Lost update: (More money)
		- happened when two transaction using the same data, they read data at the same time, but one transaction finishes after another one, data is update

but not correct.

    - Uncommitted data: (Lost money)
        - happened when two transaction using the same data, they read data at diff time, when one transaction rollback while the other one is read and done the transaction, the data is not correct.
    - Inconsistent data: ()
        - happened when one transacton is updating data while the anthoer transaction may read data before or after change
        - Serializability solve this problem
- Consurrency Control
    - What is the purpose of Locking?
        - Make sure #SPECIFIC DATA ONLY USE BY CURRENT TRANSACTION
        - Make sure transaction read #CONSISTENT DATA
    - Lock Type
        - Binary Lock : Release after transaction commit, #block reading and writing.
        - Exlusive Lock : #Only used in Writing, but will cause #DeadLock problem
        - Shared Lock : Other transaction #can Read
    - Lock Granularity(Level)
        - Database Level
            - Good for batching process
            - Bad avalibility for multi-user DBMS
        - Table Level
            - Bad avalibility for highly multi-user DBMS
        - Page Level
            - Not commonly use
        - Row Level
            - Good avalibity
            - Bad high overhead
        - Field Level
            - Good flexibility
            - Bad extremly high overhead
- Transaction Recovery Ability (Transaction logging)
    - Logging provide the aibility to restore the DB to a previous status after transaction corrupted

#3 Distributed Database
- What is distributed database?
    - A #singal logical database #physically spread across multiple computers in #multiple places connected

via network while treat as the #singal database for user.

- Distributed Database Advantages and Disadvantages:
    - Advantage[2016]
        - Fit for orgnization which #geographically distributed in diff places
        - Fast #access, #processing
        - #Easy capacity increment
        - Good #reliability, #availability, #recoverability
    - Disadvantage[2016]
        - Hard maintenance
        - More storage requirement
        - Hard grarantee data integrity, security(Replication data has high changce to be attacked)
- Two Object of Distributed DB
    - Location transparancy
        - User do not need to know where the data exactly locate in.
    - Local autonomy
        - If network failure, local DB function can work peoperly
- Two Trade-offs of Distributed DB
    - Availability VS Consistency (imagine when network fails)
    - Update Synchronous VS Update Asynchronous (if DB using replication strategy)
- [2016/7]Three Distributed Methods(Options)
    - Data Replication
        - Type:
            - [2017]Synchronous : Good for keeping up to date; Bad for high network usage
            - Asychronous : Good for response time tolerance; Bad for complex design
        - Advantages:
            - Good reliable, availability(when some nodes down)
            - Fast access
            - Low network traffic
        - Disadtanges:
            - Hard guarante data integrity
            - High demand network capacity, high sychronous time, high storage space
        - Fit for : Read-only data(Non-volatile)
    - patitioning
        - Type:
            - Horizontal Partitioning : Diff

```
rows in diff places, good in easy query
                     - Vertical Partitioning : Diff
colomns in diff places, bad in high efforts in combining
data
              - Advantages:
                     - Good efficiency & local access
optimization(close to where it is used)
                     - Good security
              - Disadvantages:
                     - Bad recoverability(No Replication
Data)
                     - Bad in inconsistent access speed


#4 Database Administration
- The role of DBA VS DA
       - DA : Plan data Policy, Standard, Solve conflict
       - DBA : Design & analyze DB, Manage security,
integrity, backup, recovery
- Database Design need to consider:
       - Calculate expected disk usage & transaction load
       - Consider the life cycle of the DB
- Database Maintenance : Monitor & Predict disk usage &
transaction load
- Capacity Planning : Table size = number of rows * Avg
rows width
       - Data type width(Bytes):
              -
Tinyint(1),SmallInt(2),MediumInt(3),Int(4), BigInt(8)
              - Float(4), Double(8)
              - Demical(4bytes/9digits)
              - Char(M), Varchar(Not static)
              - Year(1), Date/Time(3), DataTime(8),
TimeStamp(4)
              - Enum(1-2Bytes)
- How to improve Database Performance?
       - Good choice of data type, logic, index
       - Use faster storage SSD, Distributed DB
       - Caching data in memory
- Theats to DB
       - Loss of integrity : consistency & corrupted
       - Loss of availability : availble for anthorized
access
       - Loss of confidentiality : procted against
unauthorized access
-[2017] Theat Case - [SQL injection]
       - Input malicious code in input fields of web
```

application to execute the code in db query
    - How to prevent:
        - Only limited #statement & procudure
        - Give applicaiton #lease privilege
        - Check input #validation and turn all input to #ordinary characters
- DB Security Protection(How to protect DB) : Access Control & Encryption
    - Access Control
        - Give user diff levels of privilege
            - Type of discretionary privilege : Account Level, Schema Level, Table Level
        - Use VIEW (discretionary authorization mechanism)
            - hide DB structure & only provide specific data
    - Encrption
        - Table & Colomn Encrption
            - Against interception when data trasmission
            - Against unauthorized access
        - Data Encrption
            - Use encrption algorithm to encrpt data
- DB Backup
    - Just a copy of current DB, restore from copy when DB corrputed, need to plan how it backed up and how it recover
    - Why Backup? (Backup for what?)
        - Storage failure
        - Network failure
        - Statement failure
        - Process failure
        - User error
    - Type of Backup:
        - Full & Increment
            - Full : All files backup
            - Increment : Only backup changed parts
        - Online(Hot) & Offline(Cold)
            - Online(Hot) : No need shut down but need locking to ensure integrity
            - Offline(Cold) : Need shut down
        - Onsite & Offsite
            - Offsite : backup to cloud
        - Physical & Logical[2017]
            - Physical : All files raw copy;

Fast recovery; Offline; Restore: Shut down -> Copy -> Restart

        - Logical : Copy exclude log and configuration; Use query to excute; Large backup data; Slower; Online; Restore: use query
    - Backup Policy(Normal Schedule)
        - Do backup when low load
        - Prefer backup mirror db to reduce DB performance & availability
        - Test backup before restore

#5 NoSQL

-[2017] Why need NoSQL(What is constrains in Relational DB?)
    - Object Relational(OR) mismatch
    - Traditinal DB is #not good with big data
    - Traditinal DB is #not good with distributed server
- Property of NoSQL
    - #Schema-less (But maybe have #implicit schema to expect user to #input certain data)
    - #Structure-less
    - Support schema on read(data moedel is determined during data analysis process)
    - Eventually Consistent
    - Good with distributed server
    - No use relational model and SQL
    - No ACID
- Types of NoSQL
    - Key-value
        - A simple pair of a key and values. The database do not deal with structre and the meaning of values
    - Document
        - Similar to Key-value, but the document is structured so specific elements can be deal with seperately
    - Colomn-family
        - Colomn are grouped into colomn groups for efficiency purpose
    - Graph-oriented
        - Use nodes and edges stands for the relationship between data items, nodes have properties
- ACID VS BASE VS CAP Theorem
    - ACID(In Traditional DB) :
        - Atomicity
        - Consistency

- Isolation
            - Durability
        - BASE :
            - Basically Available = Guarantee the data is availale, but data maybe inconsistent
            - Soft state = System state would change even no input in because of "Eventual consistency"
            - Eventual consistency = Guarantee the data eventually being consistent once no inputs in. The data will propagate to everywhere it needs to while continuing receiving input but not check the consistency of every transaction before it moves onto the next one.
        - CAP :
            - Consistency = Data is the same after updating
            - Availability = Can Read and Write when server is working
            - Partition tolerance = System is working even two sets of server isolated
            - Trand-off between C & A in CAP (Choose one) [Leads to BASE Theorem][2016]
                - Because distributed system has to be partition tolerance, so only trade off between C & A
                - If choose Consistency
                    = If network failure, the system return error and time out because it cannot guarantee the data is up to date
                - If choose Availability
                    = If network failure, the system will continue process queries and return the lastest version of data even if it cannot guarantee that the data is up to date
- What is Big Data?
    - Large Volumn (容量大)
    - Different Variety (种类多)
    - High Velocity (速度快)
    - Schema on read(data moedel is determined during data analysis process) than Schema on write(data model is already designed)
    - Data Lake (Not follow predefined schema & flexible store and access)

#6 Trend
I- why you want to put data in the cloud?(Cloud DB advantage)
    - pay per usage

  - DBMS is adminstrated by cloud provide(reduce in-house DBA)
  - Simply setting up
  - easy scaling up
- cloud DB disadvantage
  - data security
  - legal frameworks
  - need large movements from current DB to cloud DB

I- Relational VS Object-Oriented mismatch (OO mismatch)
  - Relational: individual entities are represented as a row and based on relational.
  - OO: allow more complex objects, it reference one another and therefore form a graph in the mathematical sense. It is different from relational.

# INFO90002-Database-Systems-and-Information-Modelling
# Author: Chongzheng Zhao(https://github.com/ChongzhengZhao/)
# 2019 Semester 2