DataBase

MainLine: Designment,Development,Management

1. Conception:

- DataBase(数据库): A collection of data
- DataBase Management System & DBMS(数据库管理系统): A software
    - Massive
    - Persistent
    - Safe
    - Multi-user
    - Convenient
    - Efficient
    - Reliable
- Data Model:
    - Data design
    - 关键的entities和attributes
    - Stage: Conceptual system modeling
    - e.g. Entity Relation Diagram or UML Class Diagram
- Database Schema:
    - Database implementation
    - 每一个定义的data和relation
    - Stage: System implementation
    - e.g. Structures in DBMS: tables, colums, foreign keys etc.
- Data Dictionary:
    - 每一个定义的table和attribute
    - Define each data attribute
    - e.g. Tables of metadata
- ER Diagram
- People:
    - DBMS implementer
    - Database designer
    - Database application developer
    - Database administrator

2. Relational Model:

- 是一种基于表的数据模型
- Schema: structural decription of relation
    - Student(sno,sname,age,gender,dept)
- Instance: actual contents
    - Student(95001,"Amy",21,"M","SE")
- Some Facts:
    - Database = set of named **relations(tables)**
    - Each relation has a set of named **attributes(columns)**

    - Each **tuple(row)** has a value for each attribute
    - Each attribute has a **type(domain)**
- Key:(独特不重复的一个属性或者属性集)
    - Super Key: 所有的Key都是
    - Candidate Key: 最小不可分割的Super Key,允许为null
    - Primary Key: 人为选定的Candidate Key
    - Foreign Key: 用于外间索引的别人的Key

3. Relational Algebra
    - basic operations:
        - select σ
        - project Π
        - union $\cup$
        - set difference $-$
        - Cartesian product x
        - rename ρ
    - additional operations:
        - Set intersection $\cap$
        - Natural join $\bowtie$
        - Outer join $=\bowtie=$
        - Assignment $\leftarrow$
    - Aggregate Functions
        - avg
        - min
        - max
        - sum
        - count
    - Aggregate Operations
        - $*{G_1,...,G_n}G*{F_1(A_1),...,F_n(A_n)}(E)$
        - **Null**不参与聚集运算操作
            - $sum(a) - sum(b) \neq sum(a-b)$
            - $count(*) \neq count(a) \neq count(b)$
            - $count(name) \neq count(distinct\ name)$
4. SQL(Structured Query Language)

- DDL(Data Definition Language): **For Schema**
    - CREATE DATABASE
    - CREATE/ALTER/DROP/TRUNCATE TABLE
    - CREATE/ALTER/DROP VIEW
    - CREATE/DROP INDEX
    - data type:
        - char(n)
        - varchar(n)
        - numeric(p,d)
        - float(n)
        - Date/Time/Timestamp
        - Interval

- DML(Data Manipulation Language): **For Data**
    - Modification: INSERT, UPDATE, DELETE
    - Query: SELECT
    - data type:
        - Blob(binary large object):返回的是指针，而不是Blob
        - Clob(character large object):同上
- DCL(Data Control Language)
    - Authorization: GRANT, REVOKE
- Integrity Constraints
    - not null
    - primary key：not null + unique
    - unique
    - check(P)
    - foreign key: Referential Integrity
        - on delete/update cascade/set null/set default
    - Complex Check Clauses Like:
        - subquery in check clause
        - create assertion <assertion-name> check <predicate> not supported by anyone.
- Trigger:
    - Create Trigger name Before|After|Instead Of events [referencing-variables] [For Each Row] When (condition) action
- Procedure: 无返回值
- Function: 有返回值
- Cursor: 更新时需要显式声明for update
- API & Application
    - ODBC(Open Database Connectivity): works with C, C++, C#, and Visual Basic
    - JDBC(Java Database Connectivity): works with Java
    - Basic steps:
        1. open a connection with a database
        2. send queries and updates
        3. get back results

5. Entity_Relationship Model
6. Relational Database Design
    - Functional Dependency(FD)
        - 1NF: 属性是原子的
        - 2NF: 所有的非主属性，不能部分依赖于码
        - 3NF: 所有的非主属性，不能传递依赖于码
        - BCNF: 所有的非平凡函数依赖，应依赖于码
    - Multivalued Dependency(MVD)
        - 4NF: 对所有的非平凡多值依赖，其决定因子均来自超码
7. Transaction

    - ACID:

        - Atomicity
        - Consistency
        - Isolation

        - Durability
    - State:
        - Active: 正常运行状态
        - Partially committed: 所有语句已正常执行后
        - Failed: 异常状态
        - Aborted: 事务回滚并且数据库恢复到原始状态之后
        - Committed: 成功执行后
    - Lock
        - Shared(共享锁): 任意数量事务可同时持有
        - Exclusive(排他锁): 有一个排他锁，则任何事务不可持有任何锁
        - DeadLock(死锁): 必须有一个事务先释放锁
    - Isolation Level

Isolation Level	写锁	读锁	Dirty read	non-repeatable read	phantom	Consistency	Concurrency
1. Read uncommitted	行级排他锁，事务结束释放	不用锁	Y	Y	Y	Very Low	Very High
2. Read committed	行级排他锁，事务结束释放	行级共享锁	N	Y	Y	Low	High
3. Repeatable read	行级排他锁，事务结束释放	行级共享锁，事务结束释放	N	N	Y	High	Low
4. Serializable	表级排他锁，事务结束释放	表级共享锁，事务结束释放	N	N	N	Very High	Very Low

Problem	解释

Problem	解释
Dirty Read	读到了别人未提交的数据
Non-repeatable Read	同一事务多次读取数据值不一致，是update引发的问题
Phantom	同一事务多次读取数据数量不一致，是insert和delete引发的问题

8. Recovery System
    - Failure：
        - Transaction failure
            - Logical errors: 事务因逻辑错误无法完成
            - System errors: 由于死锁等被迫中断
        - System crash: 电源断了或者其他硬件软件故障
        - Disk failure: 磁盘损毁
    - log-based recovery
        - Deferred database modification: 直到日志commit输出后，数据库才开始执行，只需要记录新值；恢复时只redo已经commit的log
        - Immediate database modification: 在日志commit前部分完成时数据库执行，需要记录旧值和新值；恢复时遇到commit时redo，没遇到时undo
        - checkpoint优化了日志读取的效率
9. Index(优化访问速度的主要机制)
    - Ordered indices
        - **search keys** are stored in sorted order
        - Balanced trees
        - Primary index(cluster index)
        - Secondary index(non-clustering index)
        - Dense Index Files:index包含每个search-key
        - Sparse Index Files:仅仅包含一些search-key，要求search-key是顺序排列的
        - Multilevel Index
    - Hash indices: 当索引无法适应内存时，可增加多级索引提高效率
        - **search keys** are distributed in "buckets" using "hash function"
        - Hash tables
    - Covering indices: 模糊查找
10. Query Processing
    - Parsing and translation
    - Optimization
        - Equivalence Rules: 尽可能先select再join
    - Evaluation
        - Query Cost
            - disk accesses
                - Number of seeks(t_S)
                - Number of blocks read(t_T)
                - Number of blocks written(t_T)
            - CPU(ignore)
        - File scan
            - Algorithm A1(linear search)

$$cost = b_r blocktransfers + 1 seek$$

- A1(linear search, equality on key)

  $$cost = (b_r/2) * t_T + t_S$$

- Selections Using Indices
  - A2(primary index, equality on key)

    $$cost = (h_i + 1) * (t_T + t_S)$$

  - A3(primary index, equality on nonkey)

    $$cost = h_i * (t_T + t_S) + t_S + t_T * b$$

  - A4(secondary index, equality on key)

    $$cost = (h_i + 1) * (t_T + t_S)$$

  - A4(secondary index, equality on nonkey)

    $$cost = (h_i + n) * (t_T + t_S)$$

- Selections Involving Comparisons
  - A5(primary index, comparison)
  - A6(secondary index, comparison)
- Implementation of Complex Selections
  - A7(conjunctive selection using one index)
  - A8(conjunctive selection using composite index)
  - A9(conjunctive selection by intersection of identifiers)
  - A10(disjunctive selection by union of identifiers)