数据库

1 DBS&DBMS ※※

DBS：数据库系统，由数据库、DBMS、应用系统组成的数据管理系统，为了管理大量数据

DBMS：数据库管理系统，用于管理数据库中的数据，提供操作数据库的工具和接口

2 数据库系统的三级模式结构 ※※※

1. **外模式**：用户模式，数据库用户的数据视图。一个数据库可以有多个外模式
2. **模式**：逻辑模式，是数据库的全局逻辑结构和组织方式。定义了所有数据的完整视图，外模式是模式的子集。一个数据库只有一种模式
3. **内模式**：物理模式，定义数据在磁盘上的存储、索引、数据分布等

3 DBMS的主要功能 ※※

1. 数据的安全性保护
2. 数据的完整性保护
3. 并发控制
4. 数据库恢复

4 笛卡尔积和自然连接 ※※※

笛卡尔积：将多个表的**所有行组合在一起**，每个表的每一行相组合，组成所有可能组合的结果集

自然连接：基于两个表**相同列名的列进行连接操作**，把相应列值相等的行组合一起形成结果集

5 视图 ※※※

是基于基本表的**查询结果集**的临时表，通过定义视图可以更方便获取数据子集而无需编写复杂的查询语句。

- **只能查询**，不能修改。**可以简化用户操作**，是**外模式**的一种形式

- 有些视图不可更新，提供安全保护，对重构数据库提供**逻辑独立性**

6 关系数据库的完整性规则 ※※※

1. **实体完整性**：要求每个实体必须能被唯一地标识，主键必须具有**唯一性和非空性**
2. **参照完整性**：要求**引用表中的外键**和**被引用表中的主键值**相匹配
3. **用户定义的完整性**：根据具体业务需求而定义的规则

7 触发器 ※※※

触发器：**满足预定义的条件时会被触发**（对数据表进行增删改时），从而执行触发器中定义的语句集合

- 作用：强制数据完整性，数据验证和转换，记录日志，自动化业务逻辑

8 各种范式 ※※※※※※

范式是数据库设计中用于**消除数据冗余和提高数据存储效率**的规范化方法

1. **第一范式**（1BF）：**属性是不可分割的最小单元**，即没有重复列，体现原子性
2. **第二范式**：满足1NF，存在主键，**消除了非主属性的部分函数依赖**，体现唯一性
3. **第三范式**：满足2NF，非主属性互不依赖，**消除非主属性的传递依赖**
4. **BC范式**：满足3NF，确保每个非主属性**完全依赖于每个候选键**，即消除了**所有属性**对候选键的**部分函数依赖和传递依赖**

9 事务和事务的ACID特性 ※※※※※※

事务：是用户定义的数据库操作序列，这些操作要么全做要么全不做，是不可分割的工作单位

事务的ACID性质：

- **原子性**（Atomicity）：事务是一个原子操作单元，要么全成功，要么回滚到开始前的状态（发生错误）
- **一致性**（Consistency）：事务执行前后必须满足所有预定义的一致性规则（完整性约束和业务规则）
- **隔离性**（Isolation）：多个事务并发执行时，每个事务必须与其他事务隔离，互不干扰（锁&并发控制）
- **持久性**（Durability）：一旦事务被提交，所做的修改永久保存到数据库，影响是持久的（Redo & Undo）

10 封锁，三级封锁协议，两段封锁协议 ※※※※

锁：控制数据对象并发访问的机制，确保数据库操作的正确性和一致性

- **排他锁**（写锁，X锁），只能一个事务持有，阻塞其他事务对锁定对象的读和写操作
- **共享锁**（读锁，S锁），多个事务同时持有，可以并发读取但不能写入，共享锁间不互斥

三级封锁协议：讨论封锁的时机，目的是从不同程度上解决数据不一致性问题

- **一级封锁协议**：事务**修改（写入）**数据前先加X锁，**直到事务结束才释放**
- **二级封锁协议**：一级基础上，**读取**数据前先加S锁，**读完即可释放**
- **三级封锁协议**：一级基础上，**读取**数据前先加S锁，**直到事务结束才释放**

两段封锁协议：

1. **读写之前**，事务必须先获得数据的封锁
2. **释放封锁后**，事务不再申请任何封锁

11 并发操作带来的三类数据不一致性 ※※※

1. **丢失修改**（写写）：A和B同时写，会覆盖一个事务的更新
2. **不可重复读**（读写）：A读，B写，A再读数据不一样
3. **读脏数据**（写读）：A写未交，B读，若A回滚则B读到的数据无效

12 数据库恢复 ※※※

技术手段：数据转储、日志文件，以抢救丢失的数据

故障类型：

- **事务内部故障**（逻辑错误、死锁等）：利用日志文件撤销（UNDO）已经做的修改
- **系统故障**
  - 故障时事务未提交：UNDO未完成事务
  - 故障时事务已提交但尚未写回到磁盘：REDO已成功完成的事务
- **介质故障**：读取数据副本，重做所有成功事务

13 索引 ※※※※

- **聚簇索引**：就是主键索引，索引键值的逻辑顺序与表中相应行的物理顺序一致，是唯一的
- **B+树索引**：**多路平衡查找树**，查询从根节点出发，时间复杂度$O(log_m n)$，m为路数
  - 数据存储在叶节点，叶节点间有通道供平行查询
  - 不用B树？因为减少了IO次数，**查询效率更稳定**，更适合范围查找（有链表）
  - 更适用于**范围查询、排序、模糊匹配等**
- **散列索引**：**Hash表**，查询时调用Hash函数获取相应键值查询数据，时间复杂度$O(1)$
  - 更适用于**等值查询**，范围查询效果不好
  - 查询性能更好，但需要避免哈希冲突
  - 对插入、更新操作性能好$O(1)$，而B+树需要维护有序性，插入更新较慢

14 sql语句

select后面有5个子句：

1. `from`：指定数据源（表、视图）
2. `where`：基于指定条件，对数据行进行筛选
3. `group by`：基于指定条件，将数据分组
4. `having`：筛选分组
5. `order by`：对结果集排序