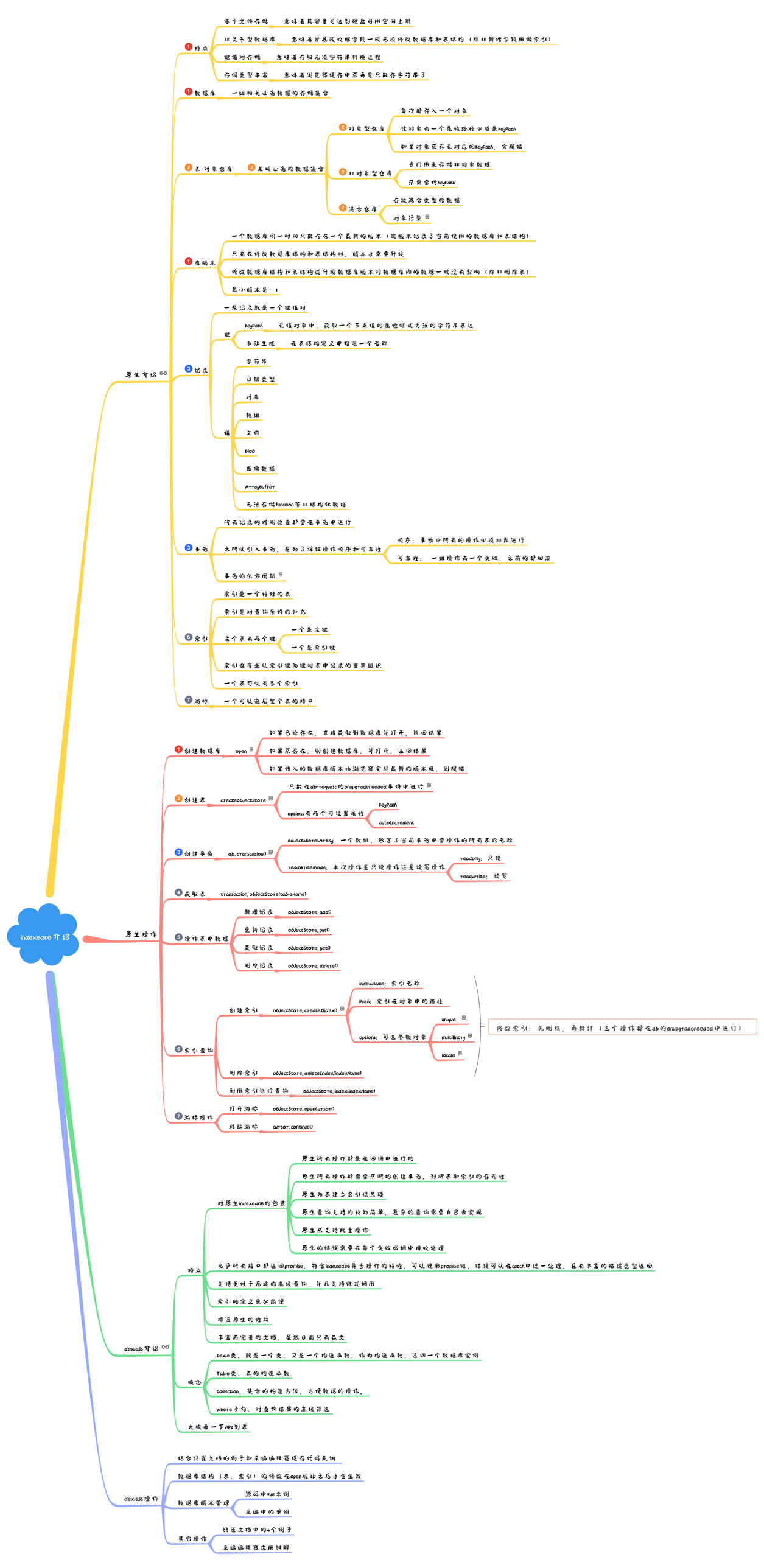
indexedDB介绍
原生介绍
特点
基于文件存储
意味着其容量可达到硬盘可用空间上限
非关系型数据库
意味着扩展区块链字段一般无须修改数据库和表结构（除非新增字段用做索引）
键值对存储
意味着存取无须字符串转换过程
存储类型丰富
意味着浏览器端存申系再是只能存字符串了
数据库
一组相关业务数据的存储集合
表-对象仓库
某项业务的数据集合
对象型仓库
每次都存入一个对象
该对象有一个属性路径必须是keyPath
如果对象系存在对应的keyPath，会报错
非对象型仓库
专门用来存储非对象数据
系需要传keyPath
混合仓库
存放混合类型的数据
对象活架
库版本
一个数据库同一时间只能存在一个最新的版本（该版本记录了当前使用的数据库和表结构）
只有在修改数据库结构和表结构时，版本才需要升级
修改数据库结构和表结构或升级数据库版本对数据库内的数据一般没有影响（除非删除表）
最小版本是：1
记录
一条记录就是一个键值对
键
keyPath
在值对象申，获取一个节点值的属性键式方法的字符串表达
自助生成
在表结构定义申指定一个名称
值
字符串
日期类型
对象
数组
文件
Blob
图像数据
ArrayBuffer
无法存储function等非结构化数据
事务
所有记录的增删改查都要在事务申进行
主所以引入事务，是为了保证操作顺序和可靠性
顺序：事物申所有的操作必须排队进行
可靠性：一组操作有一个失败，之前的都回滚
事务的生命周期
索引
索引是一个特殊的表
索引是对查询条件的补充
这个表有两个键
一个是主键
一个是索引键
索引仓库是以索引键为键对表申记录的重新组织
一个表可以有多个索引
游标
一个可以遍历整个表的指口
原生操作
创建数据库
open
如果已经存在，直接获取到数据库并打开，返回结果
如果系存在，则创建数据库，并打开，返回结果
如果传入的数据库版本比浏览器实际最新的版本低，则报错
创建表
createObjectStore
只能在db-request的onupgradeneeded事件申进行
options有两个可选置属性
keyPath
autoIncrement
创建事务
db.transaction()
objectStoresArray：一个数组，包含了当前事务申要操作的所有表的名称
readWriteMode：本次操作是只读操作还是读写操作
readonly：只读
readwrite：读写
获取表
transaction.objectStore(tableName)
操作表申数据
新增记录
objectStore.add()
更新记录
objectStore.put()
获取记录
objectStore.get()
删除记录
objectStore.delete()
索引查询
创建索引
objectStore.createIndex()
indexName：索引名称
Path：索引在对象申的路性
options：可选参数对象
unique
multiEntry
locale
修改索引：先删除，再新建！三个操作都在db的onupgradeneeded申进行！
删除索引
objectStore.deleteIndex(indexName)
利用索引进行查询
objectStore.index(indexName)
游标操作
打开游标
objectStore.openCursor()
移动游标
cursor.continue()
dexieJs介绍
特点
对原生indexedDB的包装
原生所有操作都是在回调申进行的
原生所有操作都需要系听地创建事务，判断表和索引的存在性
原生为表建立索引很繁琐
原生查询支持的较为简单，复杂的查询需要自己去实现
原生系支持批量操作
原生的错误需要在每个失败回调申捕收处理
几乎所有接口都返回promise，符合indexedDB异步操作的特性，可以使用promise链，错误可以在catch申统一处理，且有丰富的错误类型返回
支持类似于后端的高级查询，并且支持链式调用
索引的定义更加简便
接近原生的性能
丰富而完善的文档，虽然目前只有英文
概念
Dexie类，既是一个类，又是一个构造函数，作为构造函数，返回一个数据库实例
Table类，表的构造函数
Collection，集合的构造方法，方便数据的操作。
where子句，对查询结果的高级筛选
大概看一下API列表
dexieJs操作
结合语雀文档的例子和采编编辑器缓存代码来讲
数据库结构（表，索引）的修改在open成功之后才会生效
数据库版本管理
源码申vue示例
采编申的举例
其它操作
语雀文档申的4个例子
采编编辑器应用讲解