**Берілген мәтінді қазақ тіліне аударыңыз.**

Бота

一些常见的技术术语

ROVAST 2017-07-10 09:58:20 1850 收藏 1

分类专栏： 架构 文章标签： 数据库 架构

版权

前言

这里整理了自己在了解架构相关方面遇到的一些专业术语（英文），在此记录以便于以后查阅。

英文术语

C

CDN

内容分发网络（Content delivery network或Content distribution network，缩写：CDN）是指一种通过互联网互相连接的电脑网络系统，利用最靠近每位用户的服务器，更快、更可靠地将音乐、图片、视频、应用程序及其他文件发送给用户，来提供高性能、可扩展性及低成本的网络内容传递给用户。

D

DAO

在计算机软件中，数据访问对象（data access object，DAO）是为某种类型的数据库或其他持久性机制提供一个抽象接口的对象。通过映射应用程序对持久层的调用，DAO提供一些特定的数据操作，而无需暴露数据库细节。这种隔离支持单一功能原则。它分离了应用程序需要访问哪些数据，就域特定对象和数据类型（DAO的公共接口），而言，这些需求可以用数据库管理系统（DBMS）、数据库模式等满足（DAO的实现）。

尽管这种设计模式同样适用于以下内容：1、大多数编程语言；2、有持久化需求的大多数软件类型；3、大多数数据库。它在传统上与Java EE应用程序和关系数据库相关（通过JDBC API访问，因为它源于Sun Microsystems的最佳实践指南，该平台的“核心J2EE模式”）。

DBA

数据库管理员（英语：Database administrator，简称DBA），是负责管理数据库的人。数据库管理员负责在系统上运行数据库，执行备份，执行安全策略和保持数据库的完整性。因为管理数据库是个很庞大的职务，每个公司或组织的数据库管理员的需要也是很不同。一个大公司可能有很多数据库管理员，但是一个小公司可能也没有数据库管理员，而让系统管理员管理数据库。

Айдана +Мәди

DCL

数据控制语言 (Data Control Language) 在SQL语言中，是一种可对数据访问权进行控制的指令，它可以控制特定用户账户对数据表、查看表、预存程序、用户自定义函数等数据库对象的控制权。由 GRANT 和 REVOKE 两个指令组成。

DDL

数据定义语言（Data Definition Language，DDL）是SQL语言集中负责数据结构定义与数据库对象定义的语言，由CREATE、ALTER与DROP三个语法所组成，最早是由Codasyl（Conference on Data Systems Languages）数据模型开始，现在被纳入SQL指令中作为其中一个子集。

DML

数据操纵语言（Data Manipulation Language, DML）是SQL语言中，负责对数据库对象运行数据访问工作的指令集，以INSERT、UPDATE、DELETE三种指令为核心，分别代表插入、更新与删除，是开发以数据为中心的应用程序必定会使用到的指令，因此有很多开发人员都把加上SQL的SELECT语句的四大指令以“CRUD”来称呼。

DQL

数据查询语言（Data Query Language, DQL）是SQL语言中，负责进行数据查询而不会对数据本身进行修改的语句，这是最基本的SQL语句。

F

FastCGI

快速通用网关接口（Fast Common Gateway Interface／FastCGI）是一种让交互程序与Web服务器通信的协议。FastCGI是早期通用网关接口（CGI）的增强版本。

FastCGI致力于减少网页服务器与CGI程序之间交互的开销，从而使服务器可以同时处理更多的网页请求。

I

InnoDB

InnoDB，是MySQL的数据库引擎之一，为MySQL AB发行binary的标准之一。InnoDB由Innobase Oy公司所开发，2006年五月时由甲骨文公司并购。与传统的ISAM与MyISAM相比，InnoDB的最大特色就是支持了ACID兼容的事务（Transaction）功能，类似于PostgreSQL。

目前InnoDB采用双轨制授权，一是GPL授权，另一是专有软件授权。

XtraDB为派生自InnoDB的强化版，由Percona公司开发，从MariaDB的10.0.9版起取代InnoDB成为默认的数据库引擎。

Карина

M

memcached

memcached是一套分布式的高速缓存系统，由LiveJournal的Brad Fitzpatrick开发，但目前被许多网站使用。这是一套开放源代码软件，以BSD license授权发布。

memcached缺乏认证以及安全管制，这代表应该将memcached服务器放置在防火墙后。

memcached的API使用三十二比特的循环冗余校验（CRC-32）计算键值后，将数据分散在不同的机器上。当表格满了以后，接下来新增的数据会以LRU机制替换掉。由于memcached通常只是当作缓存系统使用，所以使用memcached的应用程序在写回较慢的系统时（像是后端的数据库）需要额外的代码更新memcached内的数据。

MyISAM

MyISAM是MySQL的默认数据库引擎（5.5版之前），由早期的ISAM所改良。虽然性能极佳，但却有一个缺点：不支持事务处理（transaction）。不过，在这几年的发展下，MySQL也导入了InnoDB（另一种数据库引擎），以强化参照完整性与并发违规处理机制，后来就逐渐取代MyISAM。

每个MyISAM数据表，皆由存储在硬盘上的3个文件所组成，每个文件都以数据表名称为文件主名，并搭配不同扩展名区分文件类型：

.frm－－存储数据表定义，此文件非MyISAM引擎的一部分。

.MYD－－存放真正的数据。

.MYI－－存储索引信息。

与InnoDB之比较

InnoDB可借由事务日志（Transaction Log）来恢复程序崩溃（crash），或非预期结束所造成的数据错误；而MyISAM遇到错误，必须完整扫描后才能重建索引，或修正未写入硬盘的错误。InnoDB的修复时间，大略都是固定的，但MyISAM的修复时间，则与数据量的多寡成正比。相对而言，随着数据量的增加，InnoDB会有较佳的稳定性。

MyISAM必须依靠操作系统来管理读取与写入的缓存，而InnoDB则是有自己的读写缓存管理机制。（InnoDB不会将被修改的數據頁立即交给操作系统）因此在某些情况下，InnoDB的数据访问会比MyISAM更有效率。

InnoDB目前并不支持MyISAM所提供的压缩与terse row formats，所以对硬盘与高速缓存的使用量较大。因此MySQL从5.0版开始，提供另一个负载较轻的格式，他可减少约略20%的系统负载，而压缩功能已项目于未来的新版中推出。

当操作完全兼容ACID（事务）时，虽然InnoDB会自动合并数笔连接，但每次有事务产生时，仍至少须写入硬盘一次，因此对于某些硬盘或磁盘阵列，会造成每秒200次的事务处理上限。若希望达到更高的性能且保持事务的完整性，就必使用磁盘缓存与电池备援。当然InnoDB也提供数种对性能冲击较低的模式，但相对的也会降低事务的完整性。而MyISAM则无此问题，但这并非因为它比较先进，这只是因为它不支持事务。

Нұршаш

N

NFS

网络文件系统（英语：Network File System，缩写为NFS）是一种分布式文件系统协议，最初由Sun Microsystems公司开发，并于1984年发布。[1]其功能旨在允许客户端主机可以像访问本地存储一样通过网络访问服务器端文件。

NFS和其他许多协议一样，是基于开放网络运算远程过程调用（ONC RPC）协议之上的。它是一个开放、标准的RFC协议，任何人或组织都可以依据标准实现它。

NoSql

NoSQL是对不同于传统的关系数据库的数据库管理系统的统称。

两者存在许多显著的不同点，其中最重要的是NoSQL不使用SQL作为查询语言。其数据存储可以不需要固定的表格模式，也经常会避免使用SQL的JOIN操作，一般有水平可扩展性的特征。

O

ORM

对象关系映射（英语：Object Relational Mapping，简称ORM，或O/RM，或O/R mapping），是一种程序设计技术，用于实现面向对象编程语言里不同类型系统的数据之间的转换。从效果上说，它其实是创建了一个可在编程语言里使用的“虚拟对象数据库”。如今已有很多免费和收费的ORM产品，而有些程序员更倾向于创建自己的ORM工具。

面向对象是从软件工程基本原则（如耦合、聚合、封装）的基础上发展起来的，而关系数据库则是从数学理论发展而来的，两套理论存在显著的区别。为了解决这个不匹配的现象，对象关系映射技术应运而生。

简单的说：ORM相当于中继数据。具体到产品上，例如下边的ADO.NET Entity Framework。DLINQ中实体类的属性[Table]就算是一种中继数据。

对象关系映射成功运用在不同的面向对象持久层产品中，如：Torque，OJB，Hibernate，TopLink，Castor JDO，TJDO，Active Record，NHibernate，ADO.NET Entity Framework 等。

R

Redis

Redis是一个使用ANSI C编写的开源、支持网络、基于内存、可选持久性的键值对存储数据库。从2015年6月开始，Redis的开发由Redis Labs赞助，而2013年5月至2015年6月期间，其开发由Pivotal赞助。在2013年5月之前，其开发由VMware赞助。根据月度排行网站DB-Engines.com的数据显示，Redis是最流行的键值对存储数据库。

Салсабил

中文术语

F

反向代理

在电脑网络中，反向代理是代理服务器的一种。它根据客户端的请求，从后端的服务器（如Web服务器）上获取资源，然后再将这些资源返回给客户端。与前向代理不同，前向代理作为一个媒介将互联网上获取的资源返回给相关联的客户端，而反向代理是在服务器端（如Web服务器）作为代理使用，而不是客户端。客户端通过前向代理可以访问很多不同的资源，而反向代理是很多客户端都通过它访问不同后端服务器上的资源，而不需要知道这些后端服务器的存在，而以为所有资源都来自于这个反向代理服务器。

分治法

在计算机科学中，分治法是建基于多项分支递归的一种很重要的算法范式。字面上的解释是“分而治之”，就是把一个复杂的问题分成两个或更多的相同或相似的子问题，直到最后子问题可以简单的直接求解，原问题的解即子问题的解的合并。

这个技巧是很多高效算法的基础，如排序算法（快速排序、归并排序）、傅立叶变换（快速傅立叶变换）。

另一方面，理解及设计分治法算法的能力需要一定时间去掌握。正如以归纳法去证明一个理论，为了使递归能够推行，很多时候需要用一个较为概括或复杂的问题去取代原有问题。而且并没有一个系统性的方法去适当地概括问题。

分治法这个名称有时亦会用于将问题简化为只有一个细问题的算法，例如用于在已排序的列中寻找其中一项的折半搜索算法（或是在数值分析中类似的勘根算法）。这些算法比一般的分治算法更能有效地执行。其中，假如算法使用尾部递归的话，便能转换成简单的循环。但在这广义之下，所有使用递归或循环的算法均被视作“分治算法”。因此，有些作者考虑“分治法”这个名称应只用于每个有最少两个子问题的算法。而只有一个子问题的曾被建议使用减治法这个名称。

分治算法通常以数学归纳法来验证。而它的计算成本则多数以解递回关系式来判定。

负载均衡

负载平衡（Load balancing）是一种计算机网络技术，用来在多个计算机（计算机集群）、网络连接、CPU、磁盘驱动器或其他资源中分配负载，以达到最优化资源使用、最大化吞吐率、最小化响应时间、同时避免过载的目的。

使用带有负载平衡的多个服务器组件，取代单一的组件，可以通过冗余提高可靠性。负载平衡服务通常是由专用软体和硬件来完成。

Шырын

K

空间复杂度

空间复杂度(Space Complexity)是对一个算法在运行过程中临时占用存储空间大小的量度，记做S(n)=O(f(n))。比如直接插入排序的时间复杂度是O(n2),空间复杂度是O(1) 。而一般的递归算法就要有O(n)的空间复杂度了，因为每次递归都要存储返回信息。一个算法的优劣主要从算法的执行时间和所需要占用的存储空间两个方面衡量。

N

内存数据库

内存数据库是指一种将全部内容存放在内存中，而非传统数据库那样存放在外部存储器中的数据库。内存数据库指的是所有的数据访问控制都在内存中进行，这是与磁盘数据库相对而言的，磁盘数据库虽然也有一定的缓存机制，但都不能避免从外设到内存的交换，而这种交换过程对性能的损耗是致命的。由于内存的读写速度极快（双通道DDR3-1333可以达到9300 MB/s，一般磁盘约150 MB/s），随机访问时间更是可以纳秒计（一般磁盘约10 ms，双通道DDR3-1333可以达到0.05 ms），所以这种数据库的读写性能很高，主要用在对性能要求极高的环境中，但是在服务器关闭后会立刻丢失全部储存的数据。常见的例子有MySQL的MEMORY存储引擎、eXtremeDB、FastDB、SQLite、Microsoft SQL Server Compact等。

O

耦合性

耦合性（英语：Coupling，dependency，或称耦合力或耦合度）是一种软件度量，是指一程序中，模块及模块之间信息或参数依赖的程度。

内聚性是一个和耦合性相对的概念，一般而言低耦合性代表高内聚性，反之亦然。耦合性和内聚性都是由提出结构化设计概念的赖瑞·康斯坦丁所提出。低耦合性是结构良好程序的特性，低耦合性程序的可读性及可维护性会比较好。

S

时间复杂度

在计算机科学中，算法的时间复杂度是一个函数，它定量描述了该算法的运行时间。这是一个代表算法输入值的字符串的长度的函数。时间复杂度常用大O符号表述，不包括这个函数的低阶项和首项系数。使用这种方式时，时间复杂度可被称为是渐近的，亦即考察输入值大小趋近无穷时的情况。例如，如果一个算法对于任何大小为 n （必须比 n0 大）的输入，它至多需要 5n3 + 3n 的时间运行完毕，那么它的渐近时间复杂度是 O(n3)。

为了计算时间复杂度，我们通常会估计算法的操作单元数量，每个单元运行的时间都是相同的。因此，总运行时间和算法的操作单元数量最多相差一个常量系数。

相同大小的不同输入值仍可能造成算法的运行时间不同，因此我们通常使用算法的最坏情况复杂度，记为 T(n) ，定义为任何大小的输入 n 所需的最大运行时间。另一种较少使用的方法是平均情况复杂度，通常有特别指定才会使用。时间复杂度可以用函数 T(n) 的自然特性加以分类，举例来说，有着 T(n) = O(n) 的算法被称作“线性时间算法”；而 T(n) = O(Mn) 和 Mn= O(T(n)) ，其中 M ≥ n > 1 的算法被称作“指数时间算法”。

W

微服务

微服务 (Microservices) 是一种软件架构风格，它是以专注于单一责任与功能的小型功能区块 (Small Building Blocks) 为基础，利用模组化的方式组合出复杂的大型应用程序，各功能区块使用与语言无关 (Language-Independent/Language agnostic) 的 API 集相互通讯。微服务架构运用于软件架构风格的其中一项概念是甘露运算 (Dew Computing)，意指由许多的小露水 (代表微服务的功能元件) 汇集而成的运算能力。

微服务的起源是由 Peter Rodgers 博士于 2005 年度云端运算博览会提出的微 Web 服务 (Micro-Web-Service) 开始，Juval Löwy 则是与他有类似的前导想法，将类别变成细粒服务 (granular services)，以作为 Microsoft 下一阶段的软件架构，其核心想法是让服务是由类似 Unix 管道的存取方式使用，而且复杂的服务背后是使用简单 URI 来开放界面，任何服务，任何细粒都能被开放 (exposed)。这个设计在 HP 的实验室被实现，具有改变复杂软件系统的强大力量。

2014年，Martin Fowler 与 James Lewis 共同提出了微服务的概念，定义了微服务是由以单一应用程序构成的小服务，自己拥有自己的行程与轻量化处理，服务依业务功能设计，以全自动的方式部署，与其他服务使用 HTTP API 通讯。同时服务会使用最小的规模的集中管理 (例如 Docker) 能力，服务可以用不同的编程语言与数据库等元件实作。

Алина

Z

中间件

中间件（英语：Middleware），又译中间件，是提供系统软件和应用软件之间连接的软件，以便于软件各部件之间的沟通，特别是应用软件对于系统软件的集中的逻辑，在现代信息技术应用框架如Web服务、面向服务的体系结构等中应用比较广泛。如数据库、Apache的Tomcat，IBM公司的WebSphere,BEA公司的WebLogic应用服务器，东方通公司的Tong系列中间件，以及Kingdee公司的等都属于中间件。

严格来讲，中间件技术已经不局限于应用服务器、数据库服务器。围绕中间件，Apache组织、IBM、Oracle（BEA）、微软各自发展出了较为完整的软件产品体系。（Microsoft Servers微软公司的服务器产品）。中间件技术创建在对应用软件部分常用功能的抽象上，将常用且重要的过程调用、分布式组件、消息队列、事务、安全、连结器、商业流程、网络并发、HTTP服务器、Web Service等功能集于一身或者分别在不同品牌的不同产品中分别完成。一般认为在商业中间件及信息化市场主要存在微软阵营、Java阵营、开源阵营。阵营的区分主要体现在对下层操作系统的选择以及对上层组件标准的制订。目前主流商业操作系统主要来自Unix、苹果公司和Linux的系统以及微软视窗系列。微软阵营的主要技术提供商来自微软及其商业伙伴，Java阵营则来自IBM、Sun（已被Oracle收购）、Oracle、BEA（已被Oracle收购）、金蝶（Kingdee Apusic）及其合作伙伴，开源阵营则主要来自诸如Apache，SourceForge等组织的共享代码。

中间件技术的蓬勃发展离不开标准化，标准的创建有助于融合不同阵营的系统。越来越多的标准被三大阵营共同接受并推广发展。 中间件技术的发展方向朝着更广阔范围的标准化，功能的层次化，产品的系列化方面发展。

基于中间件技术构建的商业信息软件广泛的应用于能源、电信、金融、银行、医疗、教育等行业软件，降低了面向行业的软件的开发成本。