Broadview
www.broadview.com.cn

云和恩墨
ENMOTECH

盖国强 著

Oracle DBA手记 4

数据安全警示录

修订版

云数据库时代的技能升级，超复杂数据库环境**案例精粹**

走向**自动化**、**智能化**的数据服务

安全+连续+高效+智能的解决方案实践

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

盖国强 著

Oracle DBA手记 4

数据安全警示录

修订版

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书是对众多实践案例的总结，将大量的数据安全事件、数据库安全漏洞处理案例、Oracle 数据库灾难恢复案例融合于一体，进行了详细的阐述和分析，并总结形成了指导企业规范运维、强化管理、规避灾难的指导原则。

本书通过概括总结形成了数据库运维原则，希望能够给企业运维管理者以警示，通过提前预防和规范化管理避免遭遇到书中描述的种种情形；此外，本书还通过复杂的 Oracle 数据库灾难恢复案例，深入阐述了数据库运行的内部原理，希望能为读者进一步深入探索技术提供帮助。

本书既适合负责企业自动化和智能化运维的管理者参考，也适合深入学习数据库技术的读者阅读。

图书在版编目（CIP）数据

Oracle DBA 手记 4，数据安全警示录/盖国强著. —修订本. —北京：电子工业出版社，2019.7
ISBN 978-7-121-36275-0

Ⅰ.①0… Ⅱ.①盖… Ⅲ.①关系数据库系统 Ⅳ.①TP311.138

中国版本图书馆 CIP 数据核字（2019）第 064622 号

责任编辑：刘恩惠
印　　刷：三河市良远印务有限公司
装　　订：三河市良远印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：33.75　　字数：800 千字
版　　次：2012 年 7 月第 1 版
　　2019 年 7 月第 2 版
印　　次：2019 年 7 月第 1 次印刷
定　　价：99.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

再版序言

本书第一版最初写于 2012 年，转眼已经有 8 个年头，当时还是 Oracle 10g 的天下，而现在 19c 已经发布。感谢电子工业出版社的支持，让我有机会将这本书更新补充再次出版，有一些感触在这里记录下来，奉献给支持我们的读者们。

本书主旨

这本书是我所有作品中自己最喜欢的一本，也应该是对读者和企业而言最有价值的经验分享，所以这些年来我一直念念不忘，不断记录、不断完善，希望能够在新时代让这本书再次发挥一些光和热。

借助我在职业生涯中的所见所闻，本书将那些有关数据的风险和灾难如实地刻画下来，给读者警示，引发思考，进而采取行动，制定原则，以规避和防范问题。

以我自己的职业生涯为例，我越是学习，就越是发现自己所知甚少，而所缺甚多，在飞速发展的信息技术领域，这种感觉往往使人焦虑。作为数据库或者数据环境的管理者，岗位职责非常重大，在网络上流传着很多“从删库到跑路”的故事，往往在工作中的一个不慎，就会导致不可挽回的灾难性局面，让企业遭受灭顶之灾。那么如何在这样的环境下生存，并且不负企业或者客户所托呢？

我想没有什么比在工作中不断建立指导自己工作的原则，并且坚持执行更加重要的事了。每年我们都会目睹很多因为疏忽造成的低级错误，比如误删数据文件、误删对 Oracle 生死攸关的 Redo 日志文件，这些问题和技能无关，但是事关工作原则。我曾经在一篇文章中写到，知道如何绕过问题，往往比知道如何解决问题更重要。作为 DBA 的职业素养，应该包含一种称为“预感”的能力，当我们需要执行某些任务的时候，应该敏锐地感知其中可能存在的风险点，然后设计方案进行防范或者迂回过去，而不是赴汤蹈火然后力挽狂澜。

每个人都可以承认自己的知识有限，但是不能接受缺失行动准则。风险和灾难千变万化，而规则可以以不变应对万变。

我在自己的工作历程中，不断总结提炼，形成了一系列指导自己工作的指导原则，通过这些原则的指引，我从未陷入不可控制的困境之中，也从未让自己有失所托。过去我曾经在自己的网站和一些书籍中介绍过这些

原则，其中包括“DBA 的四大守则”、“Oracle 数据库 DevOps 的三十六计”等。

本书就是我的思考和总结，这些内容不一定适合你，但我希望，通过阅读本书，读者朋友们能够思考和总结适合自己的行动原则，如果本书中记录的案例、描述的法则能够有一条让大家有感，进而发挥作用防患于未然，那就是对作者最大的回馈。

本书的第一版在内容上走了两个极端，一是通过极复杂的具体的灾难处理过程，让读者能够学习和掌握这些复杂案例的处理技能；二是通过极简的案例描述和分析，提炼可以规避问题的法则，期望以原则来防范问题。

这两个层面，也正是云和恩墨这些年走过的历程。在创业之前，云和恩墨的技术团队，以强大的单兵能力著称，屡屡临危受命力挽狂澜并且久经考验，帮助很多用户挽救了大量的数据灾难，本书中的很多案例就是这样的。然而随着服务体系的建立和完善，今天我们更加关注的是通过制定标准的服务流程和规范，通过内嵌服务理念的产品，帮助用户防患于未然，不必再陷入这样的境地。

前者治标，后者治本，显然后者才是用户真正期待的，而这些微小的技能和技巧，不过只是小道，又只有极少数的读者能够借鉴掌握并付诸实践。

所以在本书的修订过程中，我主要加强了对于后者的补充，期望在数据服务走向自动化和智能化的过程中，让这些来自实践的法则能给予读者一些帮助和助力。而本书的第 2 章正是根据这样的原则和思想高度提炼的，并早已在其他书籍中结集出版。

云数据库时代

在本书第一版出版后的 8 年时间，数据库领域已经发生了翻天覆地的变革，我以为，近代数据库技术的发展可以划分为三个阶段，分别是：

- 商业数据库时代：以 Oracle、DB2、Sybase、SQL Server 等产品为代表，开创了一个企业级软件时代。
- 开源数据库时代：以 MySQL、PostgreSQL、MongoDB、Redis 等产品为代表，推动和成就了互联网时代。
- 云数据库时代：以 Oracle 的自治数据库、AWS 的 Aurora、阿里云的 PolarDB 等产品为代表，开创了云时代。

在商业数据库时代，发展出一批伟大的产品和公司，Oracle 是其中的代表，至今仍以近 2000 亿美元的市值屹立，这个时代的商业数据库成就了一系列庞大的企业级软件，ERP、CRM 等领域都有相当体量的企业存在，这个时代也是企业级软件的时代，微软 和 IBM 都有卓越的数据库产品，曾经三足鼎立的数据库时代渐

渐发展成 Oracle 一枝独秀的态势。

在开源数据库时代，呈现出百花齐放的生态，支撑和成就了互联网行业，然而这些开源数据库产品本身并未获得应有的价值回报，近期很多开源数据库都在调整其开源协议，尤其是面对云的时代变革。历数开源时代的代表，MySQL 以 10 亿美元卖身 SUN 公司，辗转落入 Oracle 囊中，大数据的标志 Cloudera 和 Hortonworks 合并也仅仅 50 亿左右的市值，MongoDB 市值 40 亿美元左右，Elastic 市值 25 亿左右，这些数据库产品的总市值还不及微软以 262 亿美元全现金收购的 LinkedIn，这是互联网的时代。

在云数据库时代，数据库则成了云内核的一部分或一个组件，甚至不复单独存在，Amazon 提供 Redshift 和 Aurora 等多种数据库产品，阿里云有 PolarDB 等多种数据库，在这个时代，云才是永恒的主角，数据库渐渐沉淀在底层，依托云的价值而存在，AWS、微软、Google、阿里巴巴成了云时代的主角。

数据库领域经历了分分合合，再次呈现出百花齐放的局面，这让我想起《易经》的一句：见群龙无首，吉。唯有百花齐放，才见活力，企业也才有选择的自由。

诚然，大时代风起云涌，但是如果落地到企业级，我们认为用户的核心诉求仍然是不变的，云和恩墨从创业开始，就在企业手册中描述了这样一个认知，无论对于数据库还是 IT 架构，用户的核心诉求**安全、连续、高效和智能**不曾改变。所以我曾经的主题演讲用了 8 个字：稳筑基石，云帆万里。只有打好基础，才能云途高远，否则一切都只是空中楼阁，遇到真正的危机就可能轰然倒塌。

就企业的核心数据而言：

- 安全：安全是基石，尤其是云时代的数据安全，**没有数据安全，甚至就没有企业生存**，欧盟在 2018 年 5 月 25 日生效了 GDPR 法案，对安全做出了严格的要求，对违反者可以做出 2000 万欧元或企业全球年营业额 4%的重量级处罚。
- 连续：**连续运行是云时代和互联网时代的核心诉求**，无连续就无发展，传统企业也渐渐走上了 24×7 不间断服务的道路上来，数据库的连续运行只是其中的一环。
- 高效：**性能**是支持业务高速增长、获得竞争力的关键，**是成本和效率的核心**。
- 智能：**智慧智能是未来科技的必然发展方向，是企业竞争的制高点**，是产品和技术演进的终极目标，信息技术的各个环节都在走向智能化。

如前所诉，今天数据库的竞争已经转移到了云，而云时代的数据库更多的是依托前两个时代的技术积累，继承其一致性、可用性，增强了在分布式、弹性伸缩、安全方面的能力，应云而生、为云而生，谁能够在云上掌握话语权，谁才能够在未来的云时代掌握数据先机。

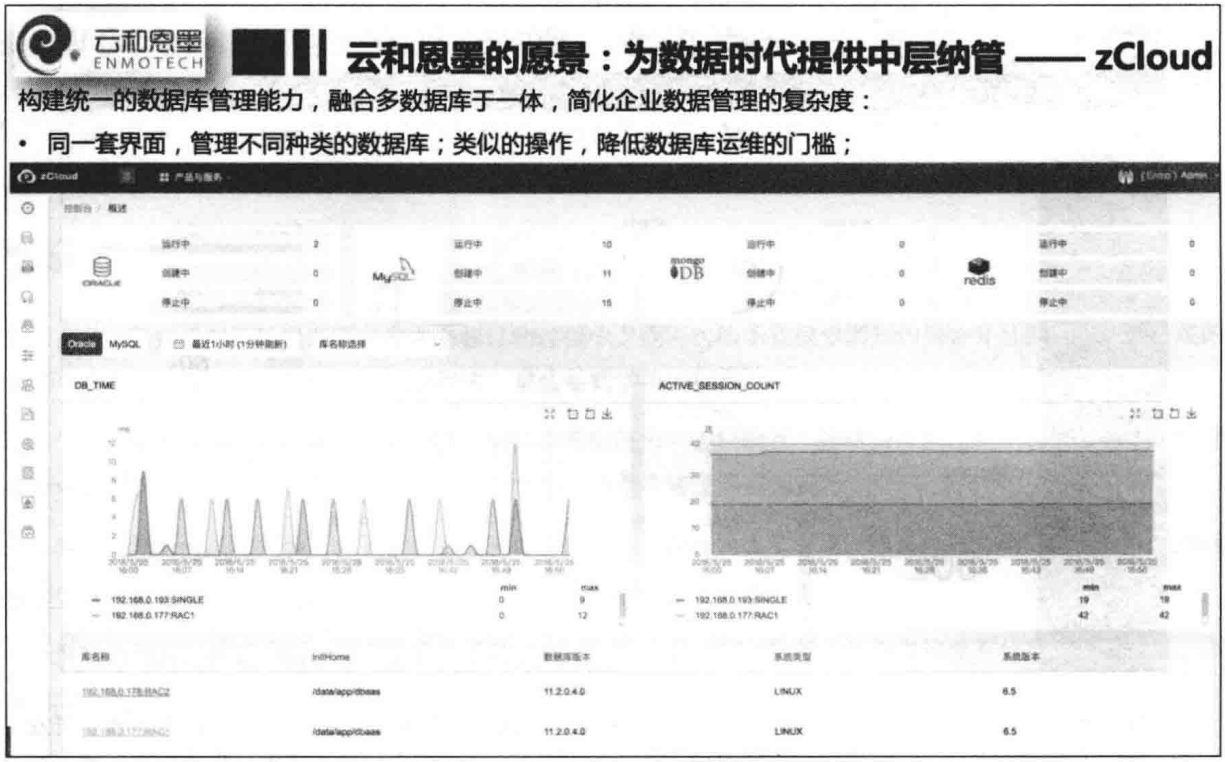
然而无论在什么时代，云上还是云下，数据安全都是企业最重要的核心诉求，云时代安全问题更加突出，这样看来，本书在今天的意义远远超越了本书第一版。

产品理念

随着百花齐放的云数据时代的到来，我们发现企业面临的数据应用和运维形态正在变得更加复杂，下图中的数据库产品往往大量存在于企业生产环境中，企业的数据库品类往往从以前的两三种，增加到今天包含商业数据库和开源数据库在内的七八种，这对企业的技术管理能力带来了巨大的挑战。

所以我认为今天的企业数据平台建设，应当改变思想，从自底向上转变为自顶向下。所谓自底向上是指，先根据业务系统的需要引入各种数据库，再向上谋求统一和简化的管理；而自顶向下则是指，先构建平台化的数据管理能力，再引入能够统一管理收缩自如的数据库产品。

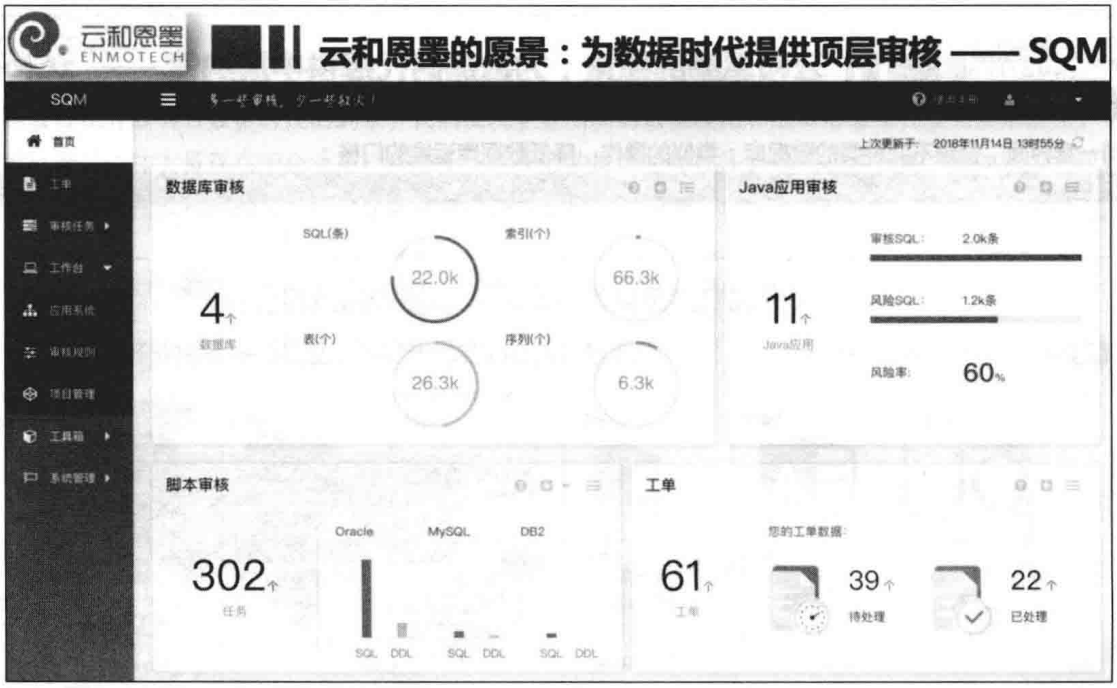
遵循数据管理平台化、统一化的思想，以及数据库运维自动化和智能化的思路，云和恩墨研发了融合多数据库管理于一体的 zCloud 云平台，只有真正具备了统一和自动化的管理能力，才能够应对云数据库时代复杂多样化的数据库环境。



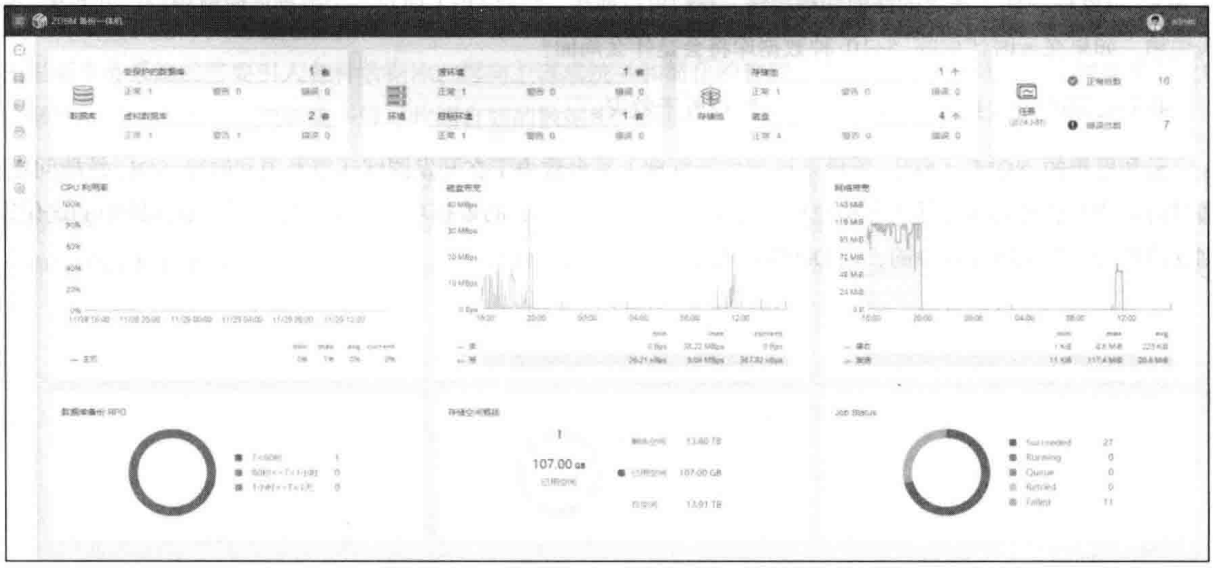
谈到数据库运维就离不开性能，而性能的核心要素是应用。在应用层业务总是通 SQL 来访问数据，控制住 SQL 也就控制住了系统的性能和稳定性，我们可以回想，以前 DBA 面对一个数据库时就曾经四处救火、焦头烂额，如果在云时代面临 5~10 种数据库将会是什么局面？

我们必须改变线上优化排障的局面，防患胜于救灾。

云和恩墨的 SQM （SQL 质量审核和性能管理）平台致力于在开发测试环节发现和解决 SQL 性能问题，通过自动化和智能化实现性能管控也就规避了线上救火。SQM 的审核功能，正是总结了来自实践中的优化原则并内置到开发测试过程中的，以 DevOps 理念实现了预防性优化。



当然，本书反复强调对于企业数据环境而言，备份重于一切，有备方能无患，所以云和恩墨独立研发了ZDBM——数据库备份一体机产品，可以通过实时的在线 Redo 备份实现近零数据丢失的数据保护，帮助用户防范意外的数据损失。



云和恩墨的产品历程，就是我们团队用过去职业生涯的知识积累、用户实践不断思考的成果输出，这些产品中涵盖了本书所陈列的各种运维安全原则，我们期望通过思考输出的作品和产品化的能力，能够帮助更多的读者和企业规避数据运维和管理风险。而这些产品中有很多都已经通过 SaaS 或其他形式免费提供给用户，包括智能巡检平台 Bethune，在线 SQL 审核和技术问答墨天轮，MySQL 数据库一体机 MyData 等等。

致谢

首先，我要感谢杨廷琨老师，在本书修订的过程中，他再次给予我很多真知灼见，并且提供了几个有趣的案例。他在技术方面的不断探究、沉稳敏锐，永远是我学习的榜样。

其次，我要感谢云和恩墨的技术团队，他们在面临压力、迎接挑战、排忧解难的过程中，践行了我们的理想和原则，不断为云和恩墨“数据驱动，成就未来”的企业理念添砖加瓦。

最后，我要感谢我的家人，虽然他们并不懂得这些技术内容的真实含义，但是他们与我共度了很多处理问题的艰难时光，给予我坚定不移的支持，并且永远是我奋斗的动力源泉。

最后的最后，我要感谢一直支持我的读者和客户们，正是你们的支持才让我继续保有分享开放的机缘，不揣浅薄，不断诉说。

但愿书中这些看起来似乎遥远的故事，能够警醒读者某些似曾相识的场景，并且永远不要面对这样的灾难。

盖国强（Eygle）

2019 年 3 月 3 日　于北京

《Oracle DBA 手记 4》序言

在数据库领域工作了十几年，我发现国内技术人员往往一直在充当救火员的角色，企业常常也认为只有能够力挽狂澜、起死回生的技术人员，才是好的技术人员。而实际上，能够不犯错误、少犯错误、提前预防、规避灾难的技术人员才是企业技术环境的最有力保障，能够未雨绸缪、防患于未然才是更好的技术实践。

我们每年都帮助很多企业挽救数据、拯救危机，本书写作的契机正是因为 2011 年 12 月 30 日和 31 日，连续两个整天，我们连续挽救了几个用户的数据库灾难，这些灾难发生的原因之简单，让人不可思议，在迈入 2012 年这个神秘年份的一刻，深深触动了我，我想，如果将这些案例描述出来，就可能帮助一些用户警醒借鉴，避免再陷入这样的困境。而**从别人的挫折中学习、借鉴，进而在自我的环境中未雨绸缪，防患于未然，是每个数据库管理人员和企业数据环境管理者应该具备的素质**。

写作这本书还和 2011 年底众多席卷而来的密码泄露事件有关，当我注视着最常用的几个密码都在互联网上被公开时，除了手忙脚乱地在各大网站修改密码，剩下的就是深深的遗憾。几乎所有从事 IT 行业的人，都深知安全的重要性，可是在实际执行中，大家又往往习惯性失明，忽视了自己本应能够达到的力所能及之安全，很多专业人士就以这样或者那样的侥幸心理忽略了风险的存在，并一步一步走向安全危机。

对于数据库安全来说，通常我们认为缺乏的并非是技术手段，更多的是缺乏规范和安全认知，如果用户都能够严格遵循安全守则并应用现有的安全技术手段，数据库的安全性就能够大幅增强，我们的安全事故率也会大大降低。

所以我决定动笔，写下自己多年来所遭遇到的安全案例以及对于数据安全的思考，如果本书中的内容能够帮助一些企业规避错误，保全数据，挽救一些技术人员的时间，那么我将感到无比欣喜，于我们的生命中，最为宝贵的就是时间，寸金难买寸光阴。

[信息安全]

在传统的信息安全领域，存在三个基本的安全要素，分别是**保密性（Confidentiality）、完整性（Integrity）和可用性（Availability），缩写为 CIA**。

这三个要素的基本定义如下：

保密性：指信息在存储、使用和传输过程中的保密性，要确保信息在这些环节中不会泄露给非授权方。

完整性：指信息在存储、使用和传输过程中不会被非授权用户篡改、变更，同时需要防止授权用户对系统及信息进行非授权篡改，保持信息在整个过程中的内外一致性。

可用性：信息系统因其服务使命，必须在用户需要时，可以被正常访问，授权用户或实体对信息系统的正常使用不应被异常拒绝或中断，应当允许其可靠、及时地访问和获取信息及资源。高可用系统要求所有时间可用，要确保系统不因电源故障、硬件故障和系统升级等因素影响服务的可用性。

信息安全的三要素是对安全的概括和提炼，不同机构和组织，因为需求不同，对 CIA 的侧重也会有所不同，随着信息安全的发展，CIA 经过细化和补充，增加了许多新的内容，包括可追溯性（Accountability）、抗抵赖性（Non-repudiation）、真实性（Authenticity）、可控性（Controllable）等。与 CIA 三元组相反的一个概念是 DAD 三元组，即泄露（Disclosure）、篡改（Alteration）和破坏（Destruction），实际上 DAD 就是信息安全面临的最普遍的三类风险，是信息安全实践活动最终应该解决的问题。

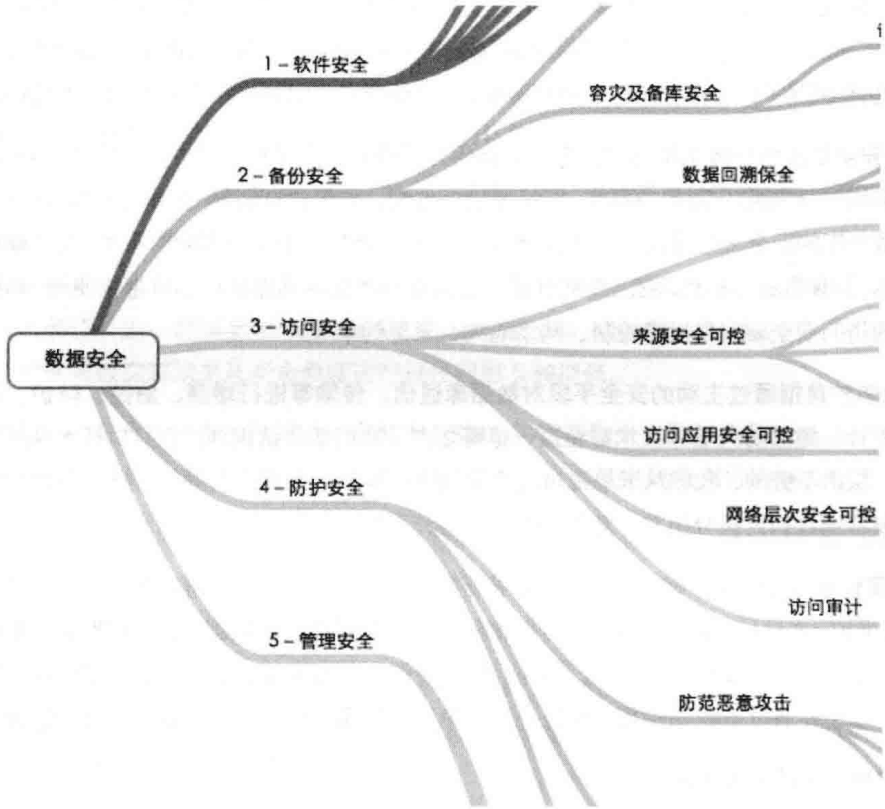
从 CIA 理念出发，我们通过对信息安全范畴所有相关主题精炼整理得到了一个标准化的知识体系，被称为公共知识体系——CBK（Common Body of Knowledge），CBK 包括 10 个知识范畴（Domain），对安全进行了全面的概括，具有极强的指导意义，这 10 个范畴分别为：访问控制，电信、网络和互联网安全，风险管理和业务连续性计划，策略、标准和组织，计算机架构和系统安全，法制、合规、调查和遵从，应用程序安全，密码学，操作安全，物理环境安全。

[数据安全]

信息安全的核心是数据安全。

在数据安全的范畴内，也包含信息安全的诸多方面，根据多年的服务经验与思考，我们将安全划分为五大方面，分别是：**软件安全、备份安全、访问安全、防护安全和管理安全。**

这五大方面是信息安全在数据领域的引申和映射，在企业数据安全中，这五大方面相辅相成、互有交叉、共同存在，下图是一张关于安全的思维导图，本书中的案例涉及了这五大方面。



在这五大安全方向中，可能出现**两种性质的安全问题，第一，由于内部管理不善而导致的数据安全问题；第二，由于外部恶意攻击入侵所带来的安全安问题**。通常我们把安全问题狭义化为后者，这实际上是片面的，在数据安全问题上，前者造成数据损失、数据损毁的概率和影响度都远远超过后者。

下面我们对数据安全的五大方面做一下简要的分析和探讨：

1. **软件安全是指数据库和应用软件产品、版本是否稳定安全；厂商所能提供的补丁集和 BUG 修正是否及时、基础硬件与操作系统是否经过认证。**很多用户在部署数据库软件时，仅仅选择了最容易获得的初始发布版本（如 Oracle Database 11.2.0.1 或者 Oracle Database 12.0.1.0 等），遗漏了可能已经存在的补丁修正，并且在运行维护中并不能够及时跟踪软件更新，也无法获得 BUG 信息、补丁修正和安全告警，这就使得软件本身的很多风险隐患得不到修正。除了数据库基础软件，如果应用软件中存在漏洞或者后门，被恶意或未授权利用，用户的数据也无法获得安全保障。如果软件安全无法保证，数据库安全的基础也就丧失了。

2. **备份安全是指用户数据能否得到及时有效的备份保全，能否在故障灾难之后获得及时的恢复和挽救。**在数据库运行期，最为重要的就是备份安全，如果没有可靠的备份，将数据集中起来就只能等待数据灾难，所以我们将备份安全提升到核心地位，备份以及随之衍生的备份加密、容灾安全、高可用、数据脱敏等，都是企

业整体数据架构应该考虑的因素。很多企业在数据灾难发生之后因为缺乏有效备份而一蹶不振，根据 Gartner 2007 年发布的一份调查报告显示，在经历了数据完全丢失而导致系统停运的企业中，有 2/5 再也没能恢复运营，余下的企业也有 1/3 在两年内宣告破产，由此可见，由于备份安全问题导致的企业伤害可能远远大于黑客攻击。

3. 访问安全是指用户数据库的访问来源和访问方式是否安全可控。通常数据库系统处于 IT 系统的核心位置，其安全架构涉及主机、系统、存储、网络等诸多方面，如果没有明确的访问控制，缺乏足够的访问分析与管理，那么数据库的安全将是混乱和无法控制的。在应用软件使用和访问数据库时，要正确设置权限，控制可靠的访问来源，保证数据库的访问安全，唯有保证访问安全才能够确保数据不被越权使用、不被误操作所损害，通常最基本的访问安全要实现程序控制、网络隔离、来源约束等。

4. 安全防护是指通过主动的安全手段对数据库通信、传输等进行增强、监控、防护、屏蔽或阻断，诸如数据加密、审计、数据防火墙等技术都在这一范畴之内。我们必须认识到，在 IT 技术高度发展的今天，风险是无处不在、层出不穷的，我们从未思考过的安全问题可能每天都在不断发生，所以在数据库环境中采取主动式防护，可以帮助我们监控分析和屏蔽很多未知风险，已经有很多成熟的产品和技术可以用于安全防范。

5. 管理安全是指在企业数据的日常管理维护范畴内，能否充分保证数据安全以及服务的高可用连续提供。诸如 DBA 的维护、文件的管理、参数或数据结构的变更等都可能引入数据风险，管理安全要求我们通过规范、制度以及技术手段去确保维护管理安全；另外，基于硬件、电力等基础平台的故障都可能影响数据库服务的高可用性，在管理中要通过监控手段及时预警，通过集群、备库等切换与服务保障服务的连续性。

这就是数据安全的五大方面。

[Oracle 数据库安全]

Oracle 数据库自 1977 年开始，就一直将安全置于首位，从强大的数据恢复机制，到不断增强的加密以及安全防范措施。“Oracle”这个名字就是来自美国中央情报局投资的项目代码，而美国中央情报局也正是 Oracle 最早期的用户之一。

接触过 Oracle 数据库的人都应当熟悉一个类似如下图所示的错误“ORA-00942：表或视图不存在”，这个简单的错误提示，最初就是在美国中央情报局的要求下作为一项安全防范设定的，这个提示的安全意义在于：**避免提供任何具体的实质性提示性信息，以预防黑客的攻击性尝试。**由此可见，安全防范可以从每一个细节入手，安全是一项全面整体的技术实现，并非孤立的存在。

```
SQL> select * from emp;
select * from emp
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

受密码事件影响，我们首先从 Oracle 数据库的密码机制上来深入了解一下 Oracle 的加密机制。虽然我们知道早在 Oracle 数据库版本 8 的年代，就已经提供了强大丰富的数据库加密功能，但是直至今日，恐怕半数以上的数据库中，仍然存放着用户的明文密码，并且未采用任何数据库安全增强机制。**这也是我认为最重要的安全问题：我们并不缺乏安全防范手段，但是缺乏对安全风险的认知**。

诚然，我们对安全的认识是随着不断出现的安全事故逐步增强的，但是希望大家都能够有计划地逐步增强对数据库的安全防范，**主动规划推进数据安全与从挫折中学习提高实有天壤之别**。对于 2011 年底的密码泄露事件，如果各大网站能够采取基本的技术手段对用户密码进行一定的加密，那么这次密码泄露的安全事件就不会显得那么初级和让人恐慌，想一想明文密码和 MD5（Message-Digest Algorithm 5，信息摘要算法 5）散列值的区别？前者基本上意味着数据库从未从安全角度进行过任何思考和增强。

Oracle 数据库的用户信息及加密密码存储于一个名为 USER$的数据表中（所有者为 SYS 用户），我们可以通过基于 USER$表建立的 DBA_USERS 视图来查询和获得这些信息，包括密码散列后的值。

在 Oracle Database 11g 之前，用户口令遵循 DES 标准（Data Encryption Standard），通过对称算法进行加密，用户名为“Salt”，密码最长为 30 个字符，所有字母被强制转换为大写。从 Oracle 7 至 Oracle 10g，加密一直使用 username 和 password 串联之后进行 HASH 运算，例如 sys/temp1 和 system/p1 将会获得相同的 HASH 加密输出。

从 Oracle Database 11g 开始，Oracle 允许最多使用 30 个字符、大小写混合方式作为密码，同时支持 DES 和 SHA-1 算法进行加密（SHA-1 算法支持大小写混合，通过初始化参数 SEC_CASE_SENSITIVE_LOGON 开关），使用 password||salt 的方式进行 HASH 加密。

下图展示了 Oracle 9i 数据库中用户密码的加密形式，DBA_USERS 视图的 PASSWORD 字段显示了加密后的密码（Encrypted Password）：

```
SQL> select username,password from dba_users
  2  where username in ('SYS','SYSTEM','EYGLE');
USERNAME                       PASSWORD
------------------------------ ------------------------------
EYGLE                          B726E09FE21F8E83
SYSTEM                         A204A4CEB2C2F2FB
SYS                            7ABF43E21B3DD9A3
```

在 Oracle 11g 中，密码从 DBA_USERS 视图中隐藏起来，这进一步增强了安全性，即便具有访问视图权限的用户，也无法获得口令的散列值（如下图所示），由此我们也可以看出 Oracle 数据库软件的安全增强历程：

```
SQL> select username,password from dba_users
  2  where username in ('SYS','SYSTEM','EYGLE');

USERNAME                       PASSWORD
------------------------------ ------------------------------
EYGLE
SYS
SYSTEM
```

密码的加密内容存储在底层的核心表（USER$是 Oracle 数据库的元数据表之一）中，以下 PASSWORD 字段存储的是 DES 标准加密值，SPARE4 存储的是 SHA-1 加密输出：

```
SQL> select * from v$version where rownum <2;

BANNER
--------------------------------------------------------------------------------
Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production

SQL> select name,password,spare4 from user$
  2  where name in ('SYS','SYSTEM','EYGLE');

NAME       PASSWORD         SPARE4
---------- ---------------- ------------------------------------------------------------------------
SYS        8A8F025737A9097A S:BBEFCBB86319E6A40372B9584DBCCA6B015BFE0C7DDF5B9593FB618E0D80
SYSTEM     2D594E86F93B17A1 S:C576FB5A54D009440AC047827392215C673528067BC06659EC56E3178BAB
EYGLE      B726E09FE21F8E83 S:65857F36842AEE4470828E9BE630FEED90A67CEF0D2B40C9FE9B558F6B49
```

关于口令的维护，Oracle 支持各种约束性限制（通过 utlpwdmg.sql 脚本启用），诸如复杂程度、长度、有效期、失败登录次数等，通过这些增强，Oracle 的口令限制可以定制出非常稳固的安全解决方案，如果你从未接触和研究过这些手段，那么可能说明你的数据库还缺乏足够的第一层安全防守。

如果我们能够从 Oracle 的安全策略入手，学习一下 Oracle 的密码安全解决方案，就能构建一套较为完善的基本安全解决方案。从 Oracle 的第一个 Internet 版本 Oracle 8i（1998 年发布）开始，Oracle 就提供了一个加密包 DBMS_OBFUSCATION_TOOLKIT 用于数据安全防护，这个加密包支持 DES、3DES 和 MD5 散列算法。

通过非常简单的封装调用，DBMS_OBFUSCATION_TOOLKIT 包就能够实现数据加密，下图展示了一个简单的示例输出，对于给定字符串进行 MD5 散列，会以 RAW 方式返回结果（通过创建稳固的函数，可以实现用户登录时的即时散列、比较和认证）：

```
SQL> set serveroutput on
SQL> BEGIN
  2    dbms_output.put_line(utl_raw.cast_to_raw(
  3    DBMS_OBFUSCATION_TOOLKIT.MD5(INPUT_STRING =>'EYGLE')));
  4  END;
  5  /
F7FB70F1E13721034765EEC4EA1A3832

PL/SQL procedure successfully completed.
```

从 Oracle Database 10g 开始，DBMS_CRYPTO 包被引入到数据库中，该程序包支持更广泛的加密算法，并用于替代 DBMS_OBFUSCATION_TOOLKIT 包，在新的版本中，诸如 DES、3DES、AES、RC4、MD5、SHA-1、MD4、HMAC_MD5、HMAC_SH1 等算法和加密方式都被支持。

通过选定的加密算法和加密方式，可以对重要数据进行加密和解密，我们不仅可以实现对密码或数值、字符数据的加密，还可以对类似 LOB 等非结构化数据进行加密。下图展示了使用 DES 算法的 CBC 模式和 PKCS5 补码规则进行加密解密的实现，示例模拟了对信用卡卡号的处理过程，金融类企业数据的安全性更为突出，需要进行安全加密的类型更为丰富。

```
SQL> set serveroutput on
SQL> DECLARE
  2    vcardno VARCHAR2(19) := '8610-1391-1812-8031';
  3    vcadraw RAW(128)     := utl_raw.cast_to_raw(vcardno);
  4    crawkey RAW(128)     := utl_raw.cast_to_raw('oracle10g');
  5    encyraw RAW(2048);
  6    decyraw RAW(2048);
  7  BEGIN
  8    dbms_output.put_line('CardNo : ' || vcardno);
  9    encyraw := dbms_crypto.encrypt(vcadraw, dbms_crypto.des_cbc_pkcs5, crawkey);
 10    dbms_output.put_line('EncdNo : ' || RAWTOHEX(utl_raw.cast_to_raw(encyraw)));
 11    decyraw := dbms_crypto.decrypt(src => encyraw, typ => dbms_crypto.des_cbc_pkcs5, KEY => crawkey);
 12    dbms_output.put_line('DecyNo : ' || utl_raw.cast_to_varchar2(decyraw));
 13  END;
 14  /
CardNo : 8610-1391-1812-8031
EncdNo : 42313246313041343030324538303234394546363245453734453341373830363245423039433745434645434631423 4
DecyNo : 8610-1391-1812-8031

PL/SQL procedure successfully completed.
```

我想重申的是，对于不同的数据库产品，都存在足够成熟的安全实现手段，应用这些安全手段就能够实现对于数据的基本保护，对于我们技术人而言，最重要的是**认识和重视数据安全问题，并逐步推动企业或组织应用安全手段进行数据安全增强**。重视数据、保护数据，这是每一位技术人的共同使命！

[本书使命]

本书所描述的所有恢复以及安全案例全部确有其事，但是基于用户隐私的考虑，我们隐去了所有和客户相关的信息，对于摘录的内容涉及用户判断的，全部进行了处理，但是时间、故障内容一切属实。多年来的灾难挽救让我积累了很多素材，所以写作这本书是一次回顾的旅程，以一条主线，将众多的灾难挽救过程串联起来。这本书中的很多案例，尚属首次曝光，而你也许还会注意到，书中的某些技术细节和恢复方法你从未在其他地方看到。

本书中的很多案例的恢复过程非常艰难，拯救过程也花费了大量的人力、物力和时间，我们也因此赢得了价值不菲的商业合同，但是从内心上来说，我们永远不希望用户陷入这样的境地，所以我写了本书，使得这些曾经惨痛的教训可以具备更广泛的借鉴意义。

基于这些想法，我对每个案例的发生过程进行了描述，并且提出了供大家警示的规避法则。所以，**我希望这是一本写给大家看的数据安全之书，不仅仅是给技术人员，更重要的是给企业数据管理者，如果不了解这些案例，你也许永远不会理解数据库为何会遭遇灭顶之灾，你也许永远无法理解为何千里之堤一朝溃于蚁穴**。

当然，这仍然是一本相当深入的技术书，我将很多案例的详细拯救过程记录下来，包括一些相当深入的技术探讨，这些技术探讨一方面可以帮助读者加深对 Oracle 数据库技术的认知，另一方面又可以在你遇到类似案例时，做出同样的营救工作。

对于我自身来说，年纪越长，就越是认识到这个世界上最为宝贵的就是时间，如果我的分享，从警示到技