Concurrency Control And Recovery In Database Systems

Database replication is widely used for fault-tolerance, scalability and performance. The failure of one database replica does not stop the system from working as available replicas can take over the tasks of the failed replica. Scalability can be achieved by distributing the load across all replicas, and adding new replicas should the load increase. Finally, database replication can provide fast local access, even if clients are geographically distributed clients, if data copies are located close to clients. Despite its advantages, replication is not a straightforward technique to apply, and there are many hurdles to overcome. At the forefront is replica control: assuring that data copies remain consistent when updates occur. There exist many alternatives in regard to where updates can occur and when changes are propagated to data copies, how changes are applied, where the replication tool is located, etc. A particular challenge is to combine replica control with transaction management as it requires several operations to be treated as a single logical unit, and it provides atomicity, consistency, isolation and durability across the replicated system. The book provides a categorization of replica control mechanisms, presents several replica and concurrency control mechanisms in detail, and discusses many of the issues that arise when such solutions need to be implemented within or on top of relational database systems. Table of Contents: Overview / 1-Copy-Equivalence and Consistency / Basic Protocols / Replication Architecture / The Scalability of Replication / Eager Replication and 1-Copy-Serializability / 1-Copy-Snapshot Isolation / Lazy Replication / Self-Configuration and Elasticity / Other Aspects of Replication

In recent years, tremendous research has been devoted to the design of database systems for real-time applications, called real-time database systems (RTDBS), where transactions are associated with deadlines on their completion times, and some of the data objects in the database are associated with temporal constraints on their validity. Examples of important applications of RTDBS include stock trading systems, navigation systems and computer integrated manufacturing. Different transaction scheduling algorithms and concurrency control protocols have been proposed to satisfy transaction timing data temporal constraints. Other design issues important to the performance of a RTDBS are buffer management, index accesses and I/O scheduling. Real-Time Database Systems: Architecture and Techniques summarizes important research results in this area, and serves as an excellent reference for practitioners, researchers and educators of real-time systems and database systems.

The last decade has brought groundbreaking developments in transaction processing. This resurgence of an otherwise mature research area has spurred from the diminishing cost per GB of DRAM that allows many transaction processing workloads to be entirely memory-resident. This shift demanded a pause to fundamentally rethink the architecture of database systems. The data storage lexicon has now expanded beyond spinning disks and RAID levels to include the cache hierarchy, memory consistency models, cache coherence and write invalidation costs, NUMA regions, and coherence domains. New memory technologies promise fast non-volatile storage and expose unchartered trade-offs for transactional durability, such as exploiting byte-addressable hot and cold storage through persistent programming that promotes simpler recovery protocols. In the meantime, the plateauing single-threaded processor performance has brought massive concurrency within a single node, first in the form of multi-core, and now with many-core and heterogeneous processors. The exciting possibility to reshape the storage, transaction, logging, and recovery layers of next-generation systems on emerging hardware have prompted the database research community to vigorously debate the trade-offs between specialized kernels that narrowly focus on transaction processing performance vs. designs that permit transactionally consistent data accesses from decision support and analytical workloads. In this book, we aim to classify and distill the new body of work on transaction processing that has surfaced in the last decade to navigate researchers and practitioners through this intricate research subject.

The latest edition of a popular text and reference on database research, with substantial new material and revision; covers classical literature and recent hot topics. Lessons from database research have been applied in academic fields ranging from bioinformatics to next-generation Internet architecture and in industrial uses including Web-based e-commerce and search engines. The core ideas in the field have become increasingly influential. This text provides both students and professionals with a grounding in database research and a technical context for understanding recent innovations in the field. The readings included treat the most important issues in the database area--the basic material for any DBMS professional. This fourth edition has been substantially updated and revised, with 21 of the 48 papers new to the edition, four of them published for the first time. Many of the sections have been newly organized, and each section includes a new or substantially revised introduction that discusses the context, motivation, and controversies in a particular area, placing it in the broader perspective of database research. Two introductory articles, never before published, provide an organized, current introduction to basic knowledge of the field; one discusses the history of data models and query languages and the other offers an architectural overview of a database system. The remaining articles range from the classical literature on database research to treatments of current hot topics, including a paper on search engine architecture and a paper on application servers, both written expressly for this edition. The result is a collection of papers that are seminal and also accessible to a reader who has a basic familiarity with database systems.

Abstract: "We focus on the update-in-place recovery mechanism for concurrency control of arbitrary operations on abstract data types. In Part I of this technical report, we consider three notions of correctness - - serial correctness, view serializability and conflict serializability. We give the definitions for recoverable, cascadefree and strict histories for arbitrary operations on objects. In Part II of this report, we study the relationship among conflict serializability, view serializability and serial correctness. For arbitrary operations on objects, we show that a sufficient condition for a history to be serially correct is that it is conflict serializable and recoverable. In Part III, we present an ordered-

shared relation called independence which goes beyond commutativity. Based on independence, we propose a concurrency control and recovery protocol that produces conflict serializable and strict histories. We have implemented our protocol and compared its performance with that of 2PL for different data contentions and resources. Our protocol performed better than 2PL for multiple disks and for medium to high data contentions."

A breakthrough sourcebook to the challenges and solutions for mobile database systems This text enables readers to effectively manage mobile database systems (MDS) and data dissemination via wireless channels. The author explores the mobile communication platform and analyzes its use in the development of a distributed database management system. Workable solutions for key challenges in wireless information management are presented throughout the text. Following an introductory chapter that includes important milestones in the history and development of mobile data processing, the text provides the information, tools, and resources needed for MDS management, including: * Fundamentals of wireless communication * Location and handoff management * Fundamentals of conventional database management systems and why existing approaches are not adequate for mobile databases * Concurrency control mechanism schemes * Data processing and mobility * Management of transactions * Mobile database recovery schemes * Data dissemination via wireless channels Case studies and examples are used liberally to aid in the understanding and visualization of complex concepts. Various exercises enable readers to test their grasp of each topic before advancing in the text. Each chapter also concludes with a summary of key concepts as well as references for further study. Professionals in the mobile computing industry, particularly e-commerce, will find this text indispensable. With its extensive use of case studies, examples, and exercises, it is also highly recommended as a graduate-level textbook.

Transaction processing is fundamental for many modern applications. These applications require the backend transaction processing engines to be available at all times as well as provide a massive horizontal scale for intensive transaction requests.Concurrency Control and Recovery features recent progress in research in online transaction processing. The book also showcases the authors' research on a highly scalable OLTP system. Its contents include the designs of an efficient multiple version storage engine, a scalable range optimistic concurrency control, high-performance Paxos-based log replication, global snapshot isolation, and fast follower recovery.This book is written for professionals, researchers, and graduate students specialising in database systems and its related fields.

This third edition of a classic textbook can be used to teach at the senior undergraduate and graduate levels. The material concentrates on fundamental theories as well as techniques and algorithms. The advent of the Internet and the World Wide Web, and, more recently, the emergence of cloud computing and streaming data applications, has forced a renewal of interest in distributed and parallel data management, while, at the same time, requiring a rethinking of some of the traditional techniques. This book covers the breadth and depth of this re-emerging field. The coverage consists of two parts. The first part discusses the fundamental principles of distributed data management and includes distribution design, data integration, distributed query processing and optimization, distributed transaction management, and replication. The second part focuses on more advanced topics and includes discussion of parallel database systems, distributed object management, peer-to-peer data management, web data management, data stream systems, and cloud computing. New in this Edition: • New chapters, covering database replication, database integration, multidatabase query processing, peer-to-peer data management, and web data management. • Coverage of emerging topics such as data streams and cloud computing • Extensive revisions and updates based on years of class testing and feedback Ancillary teaching materials are available.

With growing memory sizes and memory prices dropping by a factor of 10 every 5 years, data having a "primary home" in memory is now a reality. Main-memory databases eschew many of the traditional architectural pillars of relational database systems that optimized for disk-resident data. The result of these memory-optimized designs are systems that feature several innovative approaches to fundamental issues (e.g., concurrency control, query processing) that achieve orders of magnitude performance improvements over traditional designs. This monograph provides an overview of recent developments in main-memory database systems. It covers ?ve main issues and architectural choices that need to be made when building a high performance main-memory optimized database: data organization and storage, indexing, concurrency control, durability and recovery techniques, and query processing and compilation. The monograph focuses on four commercial and research systems: H-Store/VoltDB, Hekaton, HyPer, and SAPHANA. These systems are diverse in their design choices and form a representative sample of the state of the art in main-memory database systems. It also covers other commercial and academic systems, along with current and future research trends.

It is widely recognized by practitioners that concurrency control and recovery for transaction systems interact in subtle ways. In most theoretical work, however, concurrency control and recovery are treated as seperate, largely independent problems. In this paper we investigate the interactions between concurrency control and recovery. We consider two general recovery methods for abstract data types, update-in-place and deferred-update. While each requires operations to conflict if they do not "commute", the two recovery methods require subtly different notions of commutativity. We hive a precise characterization of the conflict relations that work with each recovery method, and show that each permits conflict relations that the other does not. Thus, the two revocery methods place incomparable constraints on concurrency control. Our analysis applies to arbitrary abstract data types, including those with operations that may be partial or non-deterministic.

Concurrency Control and Recovery in Database SystemsAddison-WesleyTransactional Information SystemsTheory, Algorithms, and the Practice of Concurrency Control and RecoveryMorgan Kaufmann

Abstract: "This paper addresses the problem of a transaction reading and writing data at multiple classification levels in a Multilevel Secure (MLS) database system. We refer to such transactions as multilevel update transactions and show that no secure scheduler can ensure atomicity of multilevel update transactions in the presence of transaction aborts. We then determine the covert channel capacity of various scheduling schemes. There are essentially two ways of scheduling multilevel update transactions. The first, which ensures strong atomicity, involves delaying the commit step of a low-level subtransaction until the fates of all siblings are known. The second scheme, which ensures only semantic atomicity, allows each subtransaction to commit or abort independently and compensates for committed transactions when necessary. Analysis of these schemes indicate that the compensation approach leads to lower covert channel bandwidths. A concurrently control and recovery protocol based on compensation is

proposed for scheduling multilevel update transactions. The correctness of the protocol is demonstrated and security issues are discussed."

High Performance MySQL is the definitive guide to building fast, reliable systems with MySQL. Written by noted experts with years of real-world experience building very large systems, this book covers every aspect of MySQL performance in detail, and focuses on robustness, security, and data integrity. High Performance MySQL teaches you advanced techniques in depth so you can bring out MySQL's full power. Learn how to design schemas, indexes, queries and advanced MySQL features for maximum performance, and get detailed guidance for tuning your MySQL server, operating system, and hardware to their fullest potential. You'll also learn practical, safe, high-performance ways to scale your applications with replication, load balancing, high availability, and failover. This second edition is completely revised and greatly expanded, with deeper coverage in all areas. Major additions include: Emphasis throughout on both performance and reliability Thorough coverage of storage engines, including in-depth tuning and optimizations for the InnoDB storage engine Effects of new features in MySQL 5.0 and 5.1, including stored procedures, partitioned databases, triggers, and views A detailed discussion on how to build very large, highly scalable systems with MySQL New options for backups and replication Optimization of advanced querying features, such as full-text searches Four new appendices The book also includes chapters on benchmarking, profiling, backups, security, and tools and techniques to help you measure, monitor, and manage your MySQL installations.

When it comes to choosing, using, and maintaining a database, understanding its internals is essential. But with so many distributed databases and tools available today, it’s often difficult to understand what each one offers and how they differ. With this practical guide, Alex Petrov guides developers through the concepts behind modern database and storage engine internals. Throughout the book, you’ll explore relevant material gleaned from numerous books, papers, blog posts, and the source code of several open source databases. These resources are listed at the end of parts one and two. You’ll discover that the most significant distinctions among many modern databases reside in subsystems that determine how storage is organized and how data is distributed. This book examines: Storage engines: Explore storage classification and taxonomy, and dive into B-Tree-based and immutable Log Structured storage engines, with differences and use-cases for each Storage building blocks: Learn how database files are organized to build efficient storage, using auxiliary data structures such as Page Cache, Buffer Pool and Write-Ahead Log Distributed systems: Learn step-by-step how nodes and processes connect and build complex communication patterns Database clusters: Which consistency models are commonly used by modern databases and how distributed storage systems achieve consistency

This book describes the theory, algorithms, and practical implementation techniques behind transaction processing in information technology systems.

We report on initial research on the concurrency control issue of compiled database applications. Such applications have a repository style of architecture in which a collection of software modules operate on a common database in terms of a set of predefined transaction types, an architectural view that is useful for the deployment of database technology to embedded control programs. We focus on decoupling concurrency control from any functionality relating to recovery. Such decoupling facilitates the compile-time query optimization. Because it is the possibility of transaction aborts for deadlock resolution that makes the recovery subsystem necessary, we choose the deadlock-free tree locking (TL) scheme for our purpose. With the knowledge of transaction workload, efficacious lock trees for runtime control can be determined at compile-time. We have designed compile-time algorithms to generate the lock tree and other relevant data structures, and runtime locking/unlocking algorithms based on such structures. We have further explored how to insert the lock steps into the transaction types at compile time. To conduct our simulation experiments to evaluate the performance of TL, we have designed two workloads. The first one is from the OLTP benchmark TPC-C. The second is from the open-source operating system MINIX. Our experimental results show TL produces better throughput than the traditional two-phase locking (2PL) when the transactions are write-only; and for main-memory data, TL performs comparably to 2PL even in workloads with many reads.

Principles of Transaction Processing is a comprehensive guide to developing applications, designing systems, and evaluating engineering products. The book provides detailed discussions of the internal workings of transaction processing systems, and it discusses how these systems work and how best to utilize them. It covers the architecture of Web Application Servers and transactional communication paradigms. The book is divided into 11 chapters, which cover the following: Overview of transaction processing application and system structure Software abstractions found in transaction processing systems Architecture of multitier applications and the functions of transactional middleware and database servers Queued transaction processing and its internals, with IBM's Websphere MQ and Oracle's Stream AQ as examples Business process management and its mechanisms Description of the two-phase locking function, B-tree locking and multigranularity locking used in SQL database systems and nested transaction locking System recovery and its failures Two-phase commit protocol Comparison between the tradeoffs of replicating servers versus replication resources Transactional middleware products and standards Future trends, such as cloud computing platforms, composing scalable systems using distributed computing components, the use of flash storage to replace disks and data streams from sensor devices as a source of transaction requests. The text meets the needs of systems professionals, such as IT application programmers who construct TP applications, application analysts, and product developers. The book will also be invaluable to students and novices in application programming. Complete revision of the classic "non mathematical" transaction processing reference for systems professionals. Updated to focus on the needs of transaction processing via the Internet-- the main focus of business data processing investments, via web application servers, SOA, and important new TP standards. Retains the practical, non-mathematical, but thorough conceptual basis of the first edition.

This book discusses action-oriented, concise and easy-to-communicate goals and challenges related to quality, reliability, infocomm technology and business operations. It brings together groundbreaking research in the area of software reliability, e-maintenance and big data analytics, highlighting the importance of maintaining the current growth in information technology (IT) adoption in businesses, while at the same time proposing process innovations to ensure sustainable development in the immediate future. In its thirty-seven chapters, it covers various areas of e-maintenance solutions, software architectures, patching problems in software reliability, preventive maintenance, industrial big data and reliability applications in electric power systems. The book reviews the ways in which countries currently attempt to resolve the conflicts and opportunities related to quality, reliability, IT and business operations, and proposes that internationally coordinated research plans are essential for effective and sustainable development, with research being most effective when it uses evidence-based decision-making frameworks resulting in clear management objectives, and is organized within adaptive management frameworks. Written by leading experts, the book is of interest to researchers, academicians, practitioners and policy makers alike who are working towards the common goal of making business operations more effective and sustainable.

Despite the growing interest in Real-Time Database Systems, there is no single book that acts as a reference to academics, professionals, and practitioners who wish to understand the issues involved in the design and development of RTDBS. Real-Time Database Systems: Issues and Applications fulfills this need. This book presents the spectrum of issues that may arise in various real-time database applications, the available solutions and technologies that may be used to address these issues, and the open problems that need to be tackled in the future. With

rapid advances in this area, several concepts have been proposed without a widely accepted consensus on their definitions and implications. To address this need, the first chapter is an introduction to the key RTDBS concepts and definitions, which is followed by a survey of the state of the art in RTDBS research and practice. The remainder of the book consists of four sections: models and paradigms, applications and benchmarks, scheduling and concurrency control, and experimental systems. The chapters in each section are contributed by experts in the respective areas. Real-Time Database Systems: Issues and Applications is primarily intended for practicing engineers and researchers working in the growing area of real-time database systems. For practitioners, the book will provide a much needed bridge for technology transfer and continued education. For researchers, this book will provide a comprehensive reference for well-established results. This book can also be used in a senior or graduate level course on real-time systems, real-time database systems, and database systems or closely related courses.
Abstract: "Shared storage arrays enable thousands of storage devices to be shared and directly accessed by end hosts over switched system-area networks, promising databases and filesystems highly scalable, reliable storage. In such systems, however, concurrent host I/Os can span multiple shared devices and access overlapping ranges potentially leading to inconsistencies for redundancy codes and for data read by end hosts. In order to enable existing applications to run unmodified and simplify the development of future ones, we desire a shared storage array to provide the illusion of a single controller without the scalability bottleneck and single point of failure of an actual single controller. In this paper, we show how rapidly increasing storage device intelligence coupled with storage's special characteristics can be successfully exploited to arrive at a high performance solution to this storage management problem. In particular, we examine four concurrency control schemes and specialize them to shared storage arrays; two centralized ones: simple server locking, and server locking with leased callbacks; and two distributed ones based on device participation: distributed locking using storage-device-embedded lock servers and timestamp ordering using loosely synchronized clocks. Simulations results show that both centralized locking schemes suffer from scalability limitations. Moreover, callback locking is particularly suspect if applications do not have much inherent locality and if the storage system introduces false sharing. Distributed concurrency control with device support is attractive as it scales control capacity with storage and performance capacity and offers the opportunity to piggyback lock/ordering messages on operation requests, eliminating message latency costs. Simulations show that both storage-optimized device-based protocols exhibit close to ideal scaling achieving 90-95% of the throughput possible under totally unprotected operation. Furthermore, timestamp ordering uses less network resources, is free from deadlocks and has performance advantages under high load. We show how timestamp ordering can be extended with careful operation history recording to ensure efficient failure recovery without inducing I/Os under normal operation. This brings the overhead of concurrency control and recovery to a negligible few percent thereby realizing the scalability potential of the shared array I/O architecture."