**CHAPTER ONE: What Is a DBA?**

**1.1 Introduction**

Data is at the center of today's applications; today's organizations simply cannot operate without data. In many ways, business today is data. Without data, businesses would not have the ability to manage finances, conduct transactions, or contact their customers. Databases are created to store and organize this data. The better the design and utility of the database, the better the organization will be positioned to compete for business.

Indeed, one of the largest problems faced by IT organizations is ensuring quality database administration. A survey of IT managers conducted by Information Week in December 2000 showed that the top two database management execution issues faced by companies are ease of administration and availability of qualified administrators.

Both of these issues were cited by 58% of survey respondents. Additionally, the 1999 Market Compensation Survey conducted by people, a Gartner Company, shows that DBA positions take longer to fill than any other position. Clearly, there is no lack of demand for DBA skills in today's job market.

**The DBA is responsible** for designing and maintaining an enterprise's databases, placing the DBA squarely at the center of the business.

The DBA has the opportunity to learn about many facets of business and how they interrelate.

The DBA can explore groundbreaking technologies as they are adopted by the organization. Exposure to new technology keeps the job stimulating but frustrating if you are trying to figure out how a new technology works for the first time.

The DBA is often working alone in these endeavors; he does not have access to additional expertise to assist when troubles arise.

* **The DBA is responsible for managing the overall database environment**. Often this includes installing the DBMS and setting up the IT infrastructure to allow applications to access databases. These tasks need to be completed before any application programs can be implemented. Furthermore, ad hoc database access is a requirement for many organizations.

Additionally, **the DBA is** in charge of setting up an ad hoc query environment that includes

* evaluating and implementing query and reporting tools,
* establishing policies and procedures to ensure efficient ad hoc queries, and
* monitoring and tuning ad hoc SQL.

As you can see, a good DBA is integral to the entire application development life cycle. The DBA is "in demand" for his knowledge of data and the way in which data is managed by modern applications.

**1.2 Database, Data, and System Administration**

Some organizations define separate roles for the business aspects and the technical aspects of data. The business aspects of data are aligned with data administration, whereas the more technical aspects are handled by database administration. Not every organization has a data administration function. Indeed, many organizations combine data administration into the database administration role.

Sometimes organizations also split up the technical aspects of data management, with the DBA responsible for using the DBMS and a system administrator or systems programmer responsible for installing and upgrading the DBMS.

**Data Administration**

Data administration separates the business aspects of data resource management from the technology used to manage data; it is more closely aligned with the actual business users of data. The data administrator (DA) is responsible for understanding the business lexicon and translating it into a logical data model. Referring back to the ADLC,

* the DA would be involved more in the requirements gathering, analysis, and design phase,
* the DBA in the design, development, testing, and operational phases.
* **Another difference between a DA and a DBA is the focus of effort.**

The DA is responsible for the following tasks:

* Identifying and cataloging the data required by business users
* Producing conceptual and logical data models to accurately depict the relationship among data elements for business processes
* Creating an enterprise data model that incorporates all of the data used by all of the organization's business processes
* Setting data policies for the organization
* Identifying data owners and stewards
* Setting standards for control and usage of data

The data administrator can be thought of as the Chief Data Officer of the corporation.

**Database Administration**

The first duty of the DBA is to understand the data models built by the DA and to communicate the model to the application developers and other appropriate technicians. The logical data model is the map the DBA will use to create physical databases. The DBA will transform the logical data model into an efficient physical database design. It is essential that the DBA incorporate his knowledge of the DBMS to create an efficient and appropriate physical database design from the logical model. The DBA should not rely on the DA for the final physical model any more than a DA should rely on a DBA for the conceptual and logical data models.

The DBA is the conduit for communication between the DA team and the technicians and application programming staff. Of course, the bulk of the DBA's job is ongoing support of the databases created from the physical design and management of the applications that access those databases. An overview of these duties is provided in the DBA Tasks section of this chapter.

The DBA is the conduit for communication between the DA team and the technicians and application programming staff.

**System Administration**

Some organizations, usually the larger ones, also have a system administrator (SA) or systems programming role that impacts DBMS implementation and operations. The SA is responsible for the installation and setup of the DBMS. The SA typically has no responsibility for database design and support. Instead, the DBA is responsible for the databases and the SA is responsible for DBMS installation, modification, and support.

The system administrator ensures that the IT infrastructure is operational for database development by setting up the DBMS appropriately, applying ongoing maintenance from the DBMS vendor, and coordinating migration to new DBMS releases and versions.

Furthermore, the SA ensures that the IT infrastructure is implemented such that the DBMS is configured to work with other enabling system software. The SA may need to work with other technicians to configure transaction processors, message queueing software, networking protocols, and operating system parameters to enable the DBMS to operate effectively. The SA ensures that the IT infrastructure is operational for database development by setting up the DBMS appropriately, applying ongoing maintenance from the DBMS vendor, and coordinating migration to new DBMS releases and versions.

As with data administration, there must be cross-training of skills between the SA and DBA. The SA will never understand the physical database like the DBA, but the DBA is unlikely to understand the installation and in-depth technical relationships of system software like the SA. Each job function will be more effective with some knowledge of the other.

If no system administration group exists, or if its focus is not on the DBMS, the DBA assumes responsibility for DBMS-related system administration and programming.

**1.3 DBA Tasks**

Ensuring that an organization's data and databases are useful, usable, available, and correct requires the DBA to perform a variety of tasks in a variety of areas. These areas include database design, performance monitoring and tuning, database availability, security, backup and recovery, data integrity, release migration—really, anything that involves the company's databases.

**Database Design**

To properly design and create relational databases, the DBA must understand and adhere to sound relational design practices. The DBA must understand both relational theory and the specific implementation of the relational database management system (RDBMS) he's using to create the database. Database design requires a sound understanding of conceptual and logical data modeling techniques. The ability to create and interpret entity-relationship diagrams is essential to designing a relational database.

The DBA must be able to transform a logical data model into a physical database implementation. The DBA must ensure that the database design and implementation will enable a useful database for the applications and clients that will use it.

Although database design is a significant skill for the DBA to possess, the job of the DBA is often disproportionately associated with database design. Although designing optimal databases is important, it is a relatively small portion of the DBA's job. A DBA will most likely spend more time administering and tuning databases than in designing and building databases.

By no means, though, should you interpret this to mean that database design is not important. A poor relational design can result in poor performance, a database that does not meet the needs of the organization, and potentially inaccurate data.

**Performance Monitoring and Tuning**

What is meant by database performance? Let's use the familiar concept of supply and demand. Users demand information from the database, and the DBMS supplies this demand for information. The rate at which the DBMS supplies the information can be termed database performance. However, it is not really that simple.

Five factors influence database performance:

* workload,
* throughput,
* resources,
* optimization, and contention.

The workload that is requested of the DBMS defines the demand. It is a combination of online transactions, batch jobs, ad hoc queries, data warehousing, analytical queries, and commands directed through the system at any given time. Workload can fluctuate drastically from day to day, hour to hour, minute to minute, and even second to second.

Throughput defines the overall capability of the computer hardware and software to process data. It is a composite of I/O speed, CPU speed, parallel capabilities of the machine, and the efficiency of the operating system and system software. The hardware and software tools at the disposal of the system are known as the resources of the system. Examples include the database kernel, disk space, cache controllers, and microcode.

Optimization refers to the analysis of database requests with query cost formulas to generate efficient access paths to data. All types of systems can be optimized, but relational queries are unique in that optimization is primarily accomplished internal to the DBMS. However, many other factors need to be optimized (SQL formulation, database parameters, programming efficiently, and so on) to enable the database optimizer to create the most efficient access paths.

When the demand (workload) for a particular resource is high, contention can result. Contention is the condition in which two or more components of the workload are attempting to use a single resource in a conflicting way (for example, dual updates to the same piece of data). As contention increases, throughput decreases.

Therefore, database performance can be defined as the optimization of resource usage to increase throughput and minimize contention, enabling the largest possible workload to be processed.

**Availability**

The availability of data and databases is often closely aligned with performance, but it is actually a separate concern. Of course, if the DBMS is offline, performance will be non existent because no data can be accessed. However, ensuring database availability is a multifaceted process.

**Database Security and Authorization**

Once the database is designed and implemented, programmers and users will need to access and modify the data. However, to prevent security breaches and improper data modification, only authorized programmers and users should have access. It is the responsibility of the DBA to ensure that data is available only to authorized users.

Typically, the DBA works with the internal security features of the DBMS in the form of SQL GRANT and REVOKE statements, as well as with any group-authorization features of the DBMS. Security must be administered for many actions required by the database environment:

* Creating database objects, including databases, tables, views, and program structures
* Altering the structure of database objects
* Accessing the system catalog • Reading and modifying data in tables
* Creating and accessing user-defined functions and data types
* Running stored procedures
* Starting and stopping databases and associated database objects
* Setting and modifying DBMS parameters and specifications
* Running database utilities such as LOAD, RECOVER, and REORG

In short, the DBA must understand and be capable of implementing any aspect of security that impacts access to databases.

**Backup and Recovery**

The DBA must be prepared to recover data in the event of a problem. "Problem" can mean anything from a system glitch or program error to a natural disaster that shuts down an organization. The majority of recoveries today occur as a result of application software error and human error. Hardware failures are not as prevalent as they used to be. In fact, analyst estimates indicate that 80% of application errors are due to software failures and human error. The DBA must be prepared to recover data to a usable point, no matter what the cause, and to do so as quickly as possible.

The majority of recoveries today occur as a result of application software error and human error.

To be prepared for any type of recovery, the DBA needs to develop a backup strategy to ensure that data is not lost in the event of an error in software, hardware, or a manual process. The strategy must be applicable to database processing, so it must include image copies of database files as well as a backup/recovery plan for database logs. It needs to account for any non-database file activity that can impact database applications, as well.

**Data Integrity**

A database must be designed to store the correct data in the correct way without that data becoming damaged or corrupted. To ensure this process, the DBA implements integrity rules using features of the DBMS. Three aspects of integrity are relevant to our discussion of databases: physical, semantic, and internal.

Most relational DBMS products provide the following types of constraints:

* Referential constraints are used to specify the columns that define any relationships between tables. Referential constraints are used to implement referential integrity, which ensures that all intended references from data in one column (or set of columns) of a table are valid with respect to data in another column of the same or a different table.
* Unique constraints ensure that the values for a column or a set of columns occur only once in a table.
* Check constraints are used to place more complex integrity rules on a column or set of columns in a table. Check constraints are typically defined using SQL and can be used to define the data values that are permissible for a column or set of columns.

The final aspect of integrity comprises internal DBMS issues. The DBMS relies on internal structures and code to maintain links, pointers, and identifiers. In most cases, the DBMS will do a good job of maintaining these structures, but the DBA needs to be aware of their existence and how to cope when the DBMS fails. Internal DBMS integrity is essential in the following areas:

* Index consistency. An index is really nothing but an ordered list of pointers to data in database tables. If for some reason the index gets out of sync with the data, indexed access can fail to return the proper data. The DBA has tools at his disposal to check for and remedy these types of errors.
* Pointer consistency. Sometimes large multimedia objects are not stored in the same physical files as other data. Therefore, the DBMS requires pointer structures to keep the multimedia data synchronized to the base table data. Once again, these pointers may get out of sync if proper administration procedures are not followed.
* Backup consistency. Some DBMS products occasionally take improper backup copies that effectively cannot be used for recovery. It is essential to identify these scenarios and take corrective actions.

**DBMS Release Migration**

The DBA is also responsible for managing the migration from release to release of the DBMS. DBMS products change quite frequently—new versions are usually released every year or so. The task of keeping the DBMS running and up- to-date is an ongoing effort that will consume many DBA cycles. Whatever approach is taken must conform to the needs of the organization, while reducing outages and minimizing the need to change applications.

The task of keeping the DBMS running and up-to- date is an ongoing effort that will consume many DBA cycles.

**Jack-of-All-Trades**

Databases are at the center of modern applications. If the DBMS fails, applications fail, and if applications fail, business can come to a halt. And if business comes to a halt often enough, the entire business can fail. Database administration is therefore critical to the ongoing success of modern business.

Databases interact with almost every component of the IT infrastructure. The IT infrastructure of today comprises many tools:

* Programming languages and environments such as COBOL, Microsoft Visual Studio, C/C++, and Java
* Database and process design tools such as Erwin and Rational Rose
* Transaction processing systems such as CICS and Tuxedo • Message queuing software such as MQSeries and MSMQ
* Networking software and protocols such as SNA, VTAM, TCP/IP, and Novell
* Networking hardware such as bridges, routers, hubs, and cabling
* Multiple operating systems such as Windows, OS/390 and MVS, UNIX, Linux, and perhaps others
* Data storage hardware and software such as enterprise storage servers, Microsoft SMS, IBM DFHSM, storage area networks (SANs), and NAS
* Operating system security packages such as RACF, ACF2, and Kerberos
* Other types of storage hardware such as tape machines, silos, and solid state (memory based) storage
* Non-DBMS data set and file storage techniques such as VSAM and B-tree
* Database administration tools
* Systems management tools and frameworks such as BMC PATROL and CA Unicenter
* Operational control software such as batch scheduling software and job-entry subsystems
* Software distribution solutions for implementing new versions of system software across the network
* Internet and Web-enabled databases and applications
* Client/server development techniques such as multitier, fat server/thin client, thin server/fat client
* Object-oriented and component-based development technologies and techniques such as CORBA, COM, OLE DB, ADO, and EJB
* PDAs such as Palm Pilots and Pocket PCs

Although it is impossible to become an expert in all of these technologies, the DBA should have some knowledge of each of these areas and how they interrelate. Even more importantly, the DBA should have the phone numbers of experts to contact in case any of the associated software and hardware causes database access or performance problems.

**1.4 New Technology and the DBA**

The DBA is at the center of the action whenever new ways of doing business and new technologies are introduced to the organization. Data is the lifeblood of modern business, data is housed by the database, and the DBA is the expert who understands database technology—and in particular, how databases can be integrated with other new technologies.

Let's examine three specific newer technologies that rely on database administration at least somewhat—to be effectively implemented: database-coupled application logic, Internet-enabled ebusiness development, and handheld computing.

**Procedural DBAs: Managing Database Logic**

Until recently, the purpose of a database management system was, appropriately enough, to store, manage, and access data. Although these core capabilities are still required of modern DBMS products, additional procedural functionality is slowly becoming not just a nice feature to have, but a necessity. Features such as triggers, user-defined functions, and stored procedures provide the ability to define business rules to the DBMS instead of in separate application programs. These features couple application logic tightly to the database server.

Since all of the most popular RDBMS products provide sometimes-complex features to facilitate database-coupled procedural logic, additional management discipline is required to ensure the optimal use of these features. Typically, as new features are added, their administration, design, and management are assigned to the DBA by default. However, without proper planning and preparation, chaos can ensue. First let's examine how database logic is stored in a DBMS.

**Stored Procedures**

Stored procedures can be thought of as programs that live in a database. The procedural logic of a stored procedure is maintained, administered, and executed through the database commands. The primary reason for using stored procedures is to move application code from a client workstation to the database server. Stored procedures typically consume less overhead in a client/server environment because one client can invoke a stored procedure that causes multiple SQL statements to be run. The alternative, the client executing multiple SQL statements directly, increases network traffic and can degrade overall application performance.

A stored procedure is a freestanding database object; it is not "physically" associated with any other object in the database. A stored procedure can access and/or modify data in many tables.

**Triggers**

Triggers are event-driven specialized procedures that are attached to database tables. The trigger code is automatically executed by the RDBMS as data changes in the database. Each trigger is attached to a single, specified table. Triggers can be thought of as an advanced form of rule or constraint that uses procedural logic. A trigger cannot be directly called or executed; it is automatically executed (or "fired") by the RDBMS as the result of a SQL INSERT, UPDATE, or DELETE statement issued on its associated table. Once a trigger is created, it is always executed when its firing event occurs.

**User-Defined Functions**

A user-defined function (UDF) provides a result based on a set of input values. UDFs are programs that can be executed in place of standard, built-in SQL scalar or column functions. A scalar function transforms data for each row of a result set; a column function evaluates each value for a particular column in each row of the results set and returns a single value. Once written, and defined to the RDBMS, a UDF becomes available just like any other built-in database function.

**1.5 The Internet: From DBA to e-DBA**

Companies of every size are using Internet technologies to speed up business processes. Indeed, ebusiness has evolved as a new term to describe the transformation of key business processes using Internet technologies. Modern organizations use the Web to communicate with their partners and customers, to connect with their back-end databases, and to conduct transactions (e-commerce). Ebusiness is the integration of traditional information technology with the Internet. This integration creates a more nimble business, prepared for the trials and tribulations of conducting business in the 21st century.

E-businesses must be able to adapt and react to constant change. When a business is online, it never closes. People expect full functionality on Web sites they visit regardless of the time. And the Web is worldwide. It may be two o'clock in the morning in New York City, but it is always prime time somewhere in the world. An e-business must be available and prepared to engage with customers 24 hours a day, 365 days a year (366 during leap years). Failure to do so risks losing business. When a Web site is down, the customer will go elsewhere to do business because the competition is just a simple mouse-click away. Therefore, those who manage an e-business must be adept, proactive, and ever vigilant.

The frantic pace of an e-business makes extreme demands on those that keep it operational, and DBAs are much affected. The need to integrate the Web with traditional Information Technology services, such as the DBMS, places high expectations on database administrators.

An e-DBA is a DBA who is capable of managing Web-based applications and their Internet-related issues. With all of the knowledge and training of a traditional DBA, the e-DBA adapts these skills to suit applications and databases that are Internet enabled. When the Web is coupled with traditional applications and databases, a complex infrastructure is the result. The e-DBA must be capable of navigating this complex, heterogeneous infrastructure and providing expertise wherever databases interact within this infrastructure.

Many factors impact database administration when you couple the Internet with database technology. Some of these issues include

* 24/7 data availability
* New technologies such as Java and XML
* Web connectivity
* Integration of legacy data with Web-based applications
* Database and application architecture
* Web-based administration
* Performance engineering for the Internet
* Unpredictable workload