

ISO 9001:2015 Certified Institute

**JAVA INSTITUTE FOR ADVANCED TECHNOLOGY**

DATABASE MANAGEMENT 1

H7DX 04

H7DX 04 /AS/01

SANITHU NAVITH GUNARATHNA

808572192V

KANDY BRANCH

**Evaluation Of DataBase Management System**

This Definitions Deals With The History and Definitions common to data-base

Technology. It delimits the objectives of data-base management systems,

Discusses important concepts, and defines terminology for use by other papers in

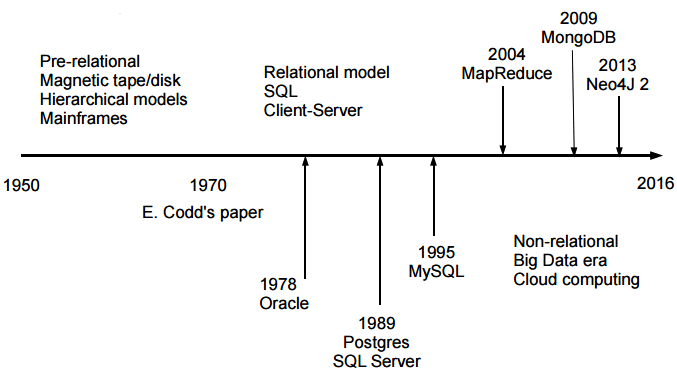
This issue, traces the development of data-base systems methodology, gives a

Uniform example, and presents some trends and issues.

Keywords and Phrases: data base, data-base management, data definition,

Data manipulation, generalized processing, data model, data independence,

Distributed Data Base, Data-Base Mach!nes, Data dictionary.



**Editor ,s 1Note:** S

Classifiation system used for references cited in

This paper.

The sorting process); those authors then proposed that these ideas be extended into other

Data - processing areas, such as file maintenance and report generation. This generalized processing entails the building of.

Special data Functions which perform frequently used, common, and repetitive Data processing tasks. But such generality cannot

Be accomplished without cost. The price of

Generalized processing is a reduction in operating efficiency, often through interpretive

Processing , or a necessary increase in resources such as hardware capacity. The success of generalized processing (and consequently of generalized data-base technology)

Thus becomes an issue of cost tradeoff.

Hardware improvements developed over

the past two decades have effected significant decreases in price/performance ratio,

thereby tending to offset operational inefficiency and to emphasize the cost of application and software development.

**CONTENTS**

1. GENERALIZED PROCESSING

2. OBJECTIVES OF DATA-BASE MANAGEMENT

Data Availability

Data Quality

Privacy and Security

Management Control

Data Independence.

3. FUNDAMENTAL CONCEPTS AND DEFINITIONS

Elements of Logical Structure

Introduction to Data Models.

Mapping from the Logical to the Physical Structure

4. HISTORICAL PERSPECTIVE

Evolution of Data Definition Languages

Centralized Data Definition: Fifties and Sixties

Stored-Data Definition Languages: 1970's

Development of Report Generator Systems

Hanford/RPG Family.

5. DEVELOPMENT OF DBMS

Early Developments: Prior to 1964

Establishment of Families: 1964-1968

Postley /Mark IV Family

Bachman/IDS Family

Formatted File/GIS Family

Vendor/CODASYL Developments: 1968 to the Present

CODASYL/DBTG Family

IMS Family

Inverted File Family

Additional Vendor Developments.

6. THE PRESIDENTIAL DATA BASE EXAMPLE

7. TRENDS AND ISSUES

Ad Hoe versus Programming Systems

Geographically Distributed Systems

Data-Base Machines

To Standardize or Not?

ACKNOWLEDGMENT

CLASSIFICATION OF REFERENCES

REFERENCES

AVAILABILITY OF REFERENCES.

Benefits of a generalized approach can thus

Be summarized as the elimination of program

Duplication (frequently found in computing

Systems), and the amortization of the onetime development costs over many applications of the Program.

In cases where a particular data-process.

1. **GENERALIZED PROCESSING**

A data-base management system (DBMS)

is a generalized tool for manipulating large

Data Bases; it is made avai.able through

Special Software for the Interrogation , maintenance, and analysis of data. Its interfaces

Generally provide a broad range of language

to aid all users--from clerk to data administrator.

DBMS technology can be traced back to

The late fifties, when authors such as McGee

[G1 and G2] 1 discussed the success of "generalized" routines. These routines were

Capable of sorting any file regardless of its

Data content (the user merely supplying

Parameters to direct the major elements of

\* This work is sponsored in part by the National

Science Foundation Grant GJ 41831.

**2. OBJECTIVES OF DATA-BASE MANAGEMENT**

The Guest Editor's Introduction to this

issue of COMPVTING SURVEYS discussed the

Concepts of data-base technology and introduced some of its objectives:

• To make an integrated collection of

Data available to a wide variety of

Users

• To provide for quality and integrity

Of the data

• To insure retention of privacy through

Security measures within the system

And

• To allow centralized control of the

Data base, which is necessary for

Efficient data administration.

To this we add the objective of "data independence," a term to be defined later [see

Page 12] in this paper. This section will deal

With each of the stated objectives, relating

Them to the overall functional architecture

of the DBMS.

While various "views of data" (the principal topic of this issue of COMPUTING SURVEYS) are important to the user interface,

The requirements for quality, integrity, security, and control have far-reaching effects.

**Data Availability**

Everest [G12] states that the major objective

of a DBMS is to make data sharing possible.

This implies that the data base as well as

programs, processes, and simulation models

are available to a wide range of users, from

the chief executive to the foreman (Everest

and Sibley [GS]). Such sharing of data reduces its average cost because the community pays for the data, while individual

users pay only for their share. However,

under these circumstances the data cannot

"belong" to any individual, program, or department; rather, it belongs to the organization as a whole.

What, then, is the overall cost of data? One

way to answer this question is by observing data entry. Keypunching and verifying,

or other types of data entry involving human keystroking, tend to cost about 50¢ per

thousand characters input. Thus, if the

average-sized file is two million characters

(a figure representative of much of today's

industry and government), it costs $1000 to

input each average-sized file. Under certain

conditions the cost of collecting data could

be substantially higher, e.g., when the data

must be collected by telemetry, or in long

and complicated experiments.

**Data Quality**

Perhaps the most neglected objective of

DBMS is the maintenance of quality. Problems relating to the quality of data and the

integrity of systems and data go hand-inhand. Data may have poor quality because

it was:

• never any good (GIGO--garbage in,

garbage out);

• altered by human error;

• altered by a program with a bug;

• altered by ~ machine error; or

• destroyed by a major catastrophe

(e.g., a mechanical failure of a disk).

Maintenance of quality involves the detection of error, determination of how the

error occurred (with preventive action to

avoid repetition of the error), and correction

of the erroneous data. These operations entail precautionary measures and additional

software functions within the data-base

management system. The prevention and

correction of the five listed causes of error

will now be briefly discussed.

In dealing with normal data-processing

applications, the programmer is faced with a

great deal of input validation. A survey by

the authors showed that about 40 % of the

PROCEDURE divisions of present-day industrial COBOL programs consists of errorchecking statements. If the validation requirements can be defined at data definition

time, then error checks may be applied automatically by the system at input, update,

manipulation, or output of data, depending

on the needs specified by the data Adminis.

**Privacy and Security**

The third major objective of data-base

management systems is privacy--the need to

protect the data base from inadvertent ac-

cess or unauthorized disclosure. Privacy is

generally achieved through some security

mechanism, such as passwords or privacy

keys. However, problems worsen when con-

trol of the system is decentralized, e.g., in

distributed data bases, where the flow of

data may overstep local jurisdictions or cross

state lines.

Who has the responsibility for the privacy

of transmitted data? When data requested

by someone with a "need to know" is put

into a nonsecure data base and subsequently

disseminated, privacy has been violated.

One solution to this problem is to pass the

privacy requirements along with the data,

which is an expensive, but necessary addi-

tion. The receiving system must then retain

and enforce the original privacy require-

ments.

Security audits, another application of the

audit trail; are achieved by logging access

(by people and programs) to any secure in-

formation. These mechanisms allow a se-

curity officer to determine who has been ac-

cessing what data under what conditions,

thereby monitoring possible leakage and pre-

venting any threat to privacy. Much of this

technology is, however, still in its infancy.

**Management Control**

The need for management control is central

to the objectives of data-base management.

It includes the establishment of the data ad-

ministration function and the design of

effective data bases. Data administration

currently uses primitive tools; a discussion of

them would be beyond the scope of this

paper (see [A1, 2, and 3]). However, it is

important to note that data-base design in-

volves tradeoffs, because users may have

quite incompatible requirements. As an ex-

ample, one group may require very rapid

response to ad hoc requests, while another

requires long and complicated updating

with good security and quality control of

the data. The implementation of a system

responsive to the first need may suggest a

storage technique quite different from that

needed by the second. The only way to re-

solve such a conflict is to determine which

user has the major need. If the requirements

are equally important, a duplicate data base

may be necessary--one for each class of user.

Although the installation of a data-base

management system is an important step to-

ward effective management control, today's

data administrator faces a challenge: the

available tools are simplistic and seldom

highly effective. They involve simulation,

data gathering, and selection techniques.

Some new analytical methods appear promis-

ing [G3]. These methods select the "best"

among several options of storage techniques,

but they are usually associated with one

particular DBMS rather than with several.

**Data Independence**

Many definitions have been offered for the

term data independence, and the reader

should be aware that it is often used am-

biguously to define two different concepts.

But first, we must define other terms. A

physical structure2 describes the way data

values are stored within the system. Thus

pointers, character representation, floating-

point and integer representation, ones- or

2 T h e t e r m s data structure a n d storage structure,

which were promulgated by the CODASYL Sys-

tems Committee [U2] can be attributed to l)'Im-

perio [DL2]. However, in computer science, the

term data structure is more closely associated with

physical implementation techniques such as linked

lists, stacks, ring structures, etc. To prevent am-

biguity we opt for the more basic terms, logical

and physical structure.

**3. FUNDAMENTAL CONCEPTS AND**

**DEFINITIONS**

Some important ideas were introduced when

we discussed the basic objectives of DBMS.

This section presents further concepts and

definitions.

Unfortunately, our language is rich in its

words and semantics about data. Entity,

item, name, element, value, instance, and

occurrence (to name a few) come ready-

equipped with meaning, yet they are used in

different ways. We must be precise, and are

thus forced to make exact definitions for

these words which we must use consistently.

Elements of Logical Structure

The starting point is to define the object of

the discourse, the entity, and the process of

its definition, which is a modeling process.

A human being is constantly "modeling" in-

formation--a baby sees an animal and says

"dog" (though it may be a horse). The

process of modeling information as data

often involves trial-and-e~ror. First, infor-

mation needs are determined, next data

(and processes) are structured to satisfy the

needs, and then data is restructured because

of changes in the need or necessary improve-

ments to the model.

The principal construct in the data struc-

turing process is the entity:

An information system deals with objects and

events in the real world that are of interest. These

real objects and events, called entities, are repre-

sented in the system by data.

**Introduction to Data Models**

The evolving field of data models is often

hotly debated. Proponents of each model

point out its advantages, but so far there is

no concensus as to the best version. In

reality, there is a spectrum of data models

ranging from the CoBoL-like "fiat file"

(single entity model) to the complex ex-

tended-set model.

Since COBOL, the most widely used lan-

guage today, has a DATA DIVISION with

data definition capabilities, it represents a

good starting point for the discussion of data

models. Though limited, this data definition

capability allows tile group (termed a

RECORD in CoBoL) to be defined as an 01

level, followed by the items, groups, and re-

peating groups at other levels. The PRESI-

D E N T entity, discussed previously, is shown

in Figure 2.

In COBOL, each item is formatted by de-

O1 PRESIOENT.

02 PRES-NAME PICTURE X ( 2 0 ) . . .

02 SPOUSE PIC X IIO)...

02 BIRTH- DATE. ,.

05 MONTH...

O3 DAY PIC..,

O3 YEAR...

02 CHILOREN OCCURS O TO N TIMES.,.

05 C - N A M E . . .

0:3 DATE - OF - BIRTH...

FIGURE 2. A CoBoL-like definition for

PRESIDENT

18 • James P. Fry and Edgar H. Sibley

Convegt Relational Nawork Hierarchic

I t e m

Item value

Group.

Entity (type)

Entity instance

Relationship

Relationship

instance

Data administrator

view

Definition of data

administrator view

User view

Definition Of user

view

Data-base subdivi-

sion

Entry points

Single u n i q u e / i d e n t i -

fier

role name/domain •-data item type item/field

component . . . . • data item occurrence value

not affowed. " igroup group

relation, record type entry/segment type

tuple record occurrence entry/segment occur-

r e n c e

foreign key comparable set type hierarchic (implied)

underlying domains

set occurrence assembly

d a t a model logical s t r u c t u r e logical s t r u c t u r e

d a t a model definition schema schema

d a t a submodel

d a t a submodel defini- s u b s c h e m a

tion

area

subschema

p r i m a r y key

c a n d i d a t e key

singular sets CALC records

key

root group

root s e g m e n t

sequencer (unique)

TABLE 1. COMPARATIVE TECHNOLOGY.

**Advantages and Disadvantages of the Database Management Systems**

* **Advantage**

1. Imporved Data Sharing.

* The DBMS helps create an environment in which end users have better access to more and

better-managed data.

1. Improved Data Security

* The more users access the data, the greater the risks of data security breaches.Corporations invest considerable amounts of time, effort, and money to ensure that corporate data are used properly

1. Better data integration

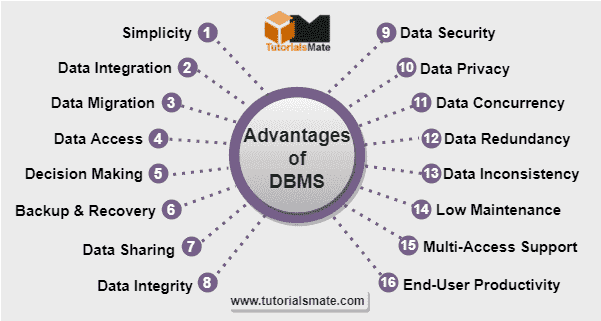
* Wider access to well-managed data promotes an integrated view of the organization’s operations and a clearer view of the big picture.

1. Minimized data inconsistency:

* Data inconsistency exists when different versions of the same data appear in different places.

1. Improved data access:

* The DBMS makes it possible to produce quick answers to ad hoc queries.



* **Disadvantages**

1. Increased costs:

* Database systems require sophisticated hardware and software and highly skilled personnel.

1. Management complexity:

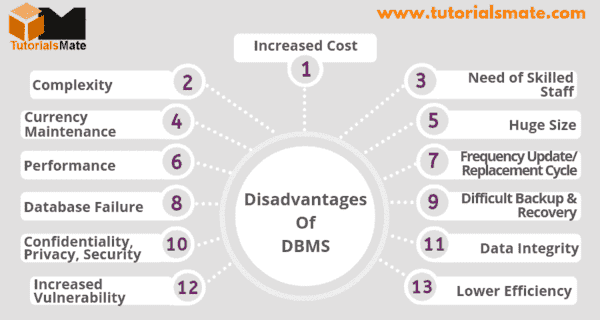
* Database systems interface with many different technologies and have a significant impact on a company’s resources and culture.

1. Maintaining currency:

* To maximize the efficiency of the database system, you must keep your system current.

1. Frequent upgrade/replacement cycles:

* DBMS vendors frequently upgrade their products by adding new functionality. Such new features often come bundled in new upgrade versions of the software.

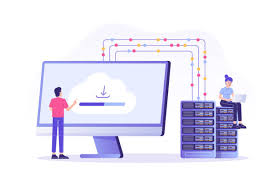


1. **The Job Roles Of Database Management System**

* **Data Administrator (DA)**
* **Database Administrator (DBA)**
* **Database Designer**
* **Application Designer**
* **End User**

**Data Administrator (DA)**

A database administrator, or DBA, is responsible for maintaining, securing, and operating databases and also ensures that data is correctly stored and retrieved. In addition, DBAs often work with developers to design and implement new features and troubleshoot any issues.

**** 

**Data Administrator Main Functions**

Functions of Data Administrator in DBMS

The data administrator plays an important role in the functioning of the database in DBMS; the fundamental role of the data administrator in DBMS is to support, apply or execute, control, retrieve, and regress or optimize the database for an organization.

**Data Administrators Salary**

The Average Salary of a Data Administrator is **$131,342** and Their Salary Range is Min: **$70K Max: $260K**.

**Highest Paying Countries in Need of Data Administrator 2022**

|  |  |  |
| --- | --- | --- |
| Country | Salary Category | Salary **($)** |
| United States | Average Annual Salary | $165,000 |
| Switzerland | Average Annual Salary | $140,000 |
| UK | Average Annual Salary | $120,000 |
| Australia | Average Annual Salary | $124,000 |

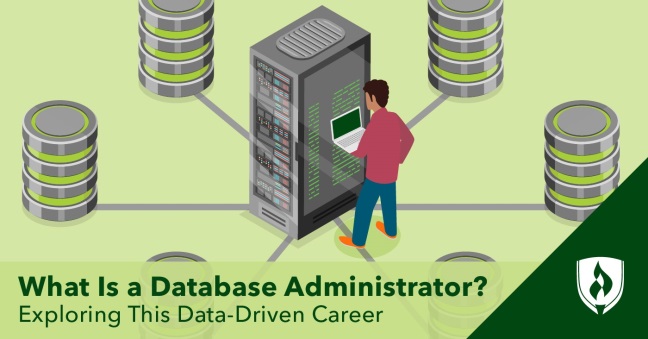
**Data Administrator Salary Range Year By Year**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Job Title | 2020 Salary | 2021 Salary | 2022 Salary | 2-Year Change |
| Data Administrator | $79,250 - $120,250 | $79,750 - $120,500 | $82,750 - $124,500 | + 3.9% |

**Database Administrator (DBA)**

A database administrator, or DBA, is responsible for maintaining, securing, and operating databases and also ensures that data is correctly stored and retrieved.

** **

 ****

In addition, DBAs often work with developers to design and implement new features and troubleshoot any issues. A DBA must have a strong understanding of both technical and business needs.

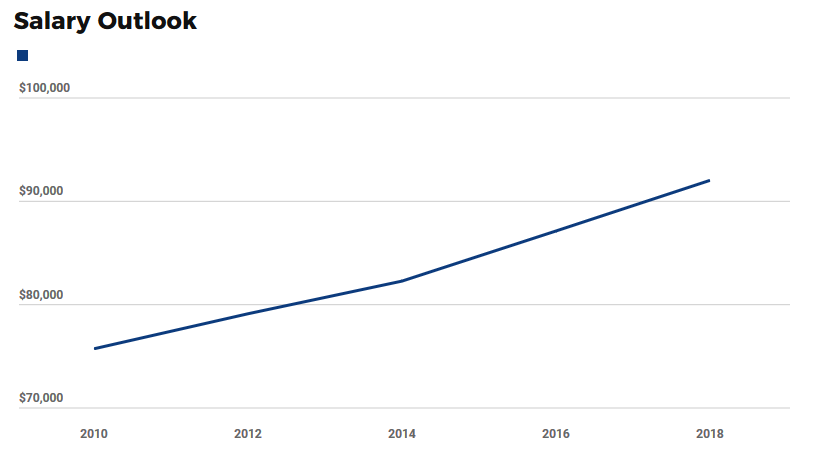
The role of DBA is becoming increasingly important in today’s information-driven business environment. Thoroughout the world, more and more organizations depend on data to discover analytical insights on market conditions, new business models, and cost-cutting measures. The global cloud computing market is also expected to expand as companies move their business operations to the cloud. Consequently, the need for qualified DBAs will only continue to grow.

**Database Administrator Main Functions**

Main Function of DBAs Manage database deployment topologies, data models, and application access patterns. They are responsible for capacity planning and infrastructure provisioning. DBAs work with developers to ensure that data models support application requirements.

**Database Administrator Salary**

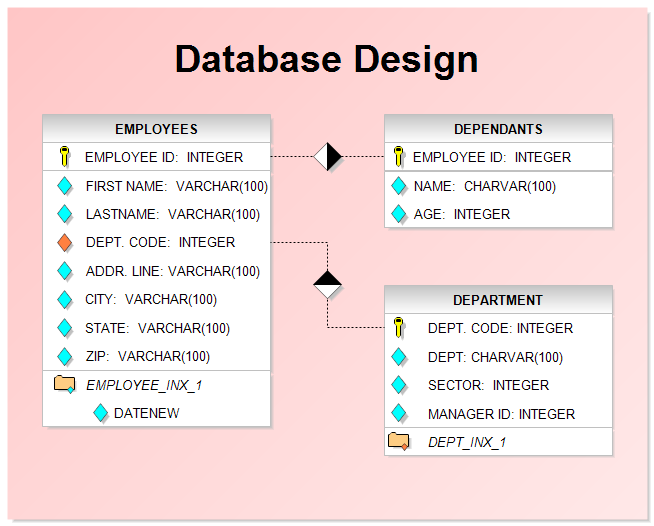
The average salary for a Database Administrator with 7+ years of experience is **$138,682**. The average salary for <1 year of experience is **$108,500**.



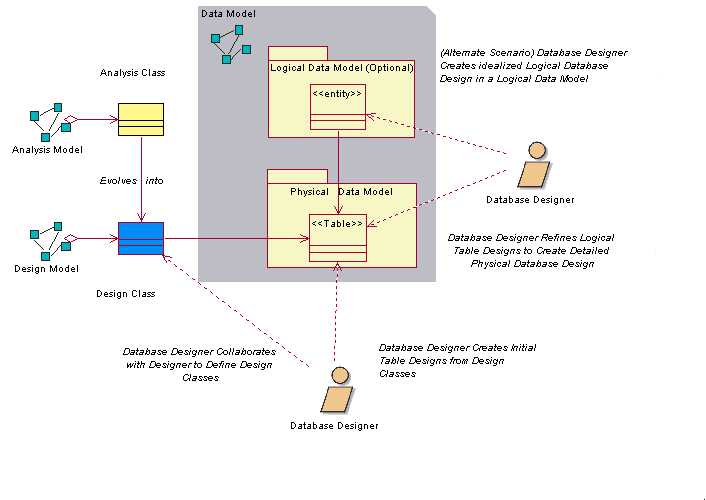
2010 – 2018 Salary Outlook (Database Administrators) …

**Database Designer**

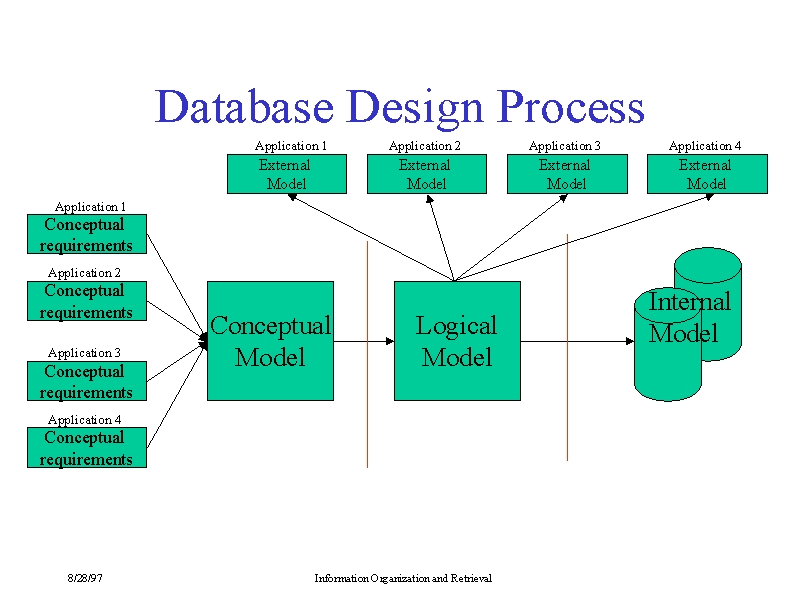
To find out the answer to the question, 'What is a database designer?', you can look at their varied daily tasks that can range from creating to maintaining an organisation's database to facilitate their work efficiency. A database designer is in charge of designing, developing, executing and preserving a company's data management systems. One of the most important responsibilities of a database designer is to form relationships between various elements of data and give it a logical structure.

** **

**How to Design a Database By a Database Designer**

****

**Database Design Process**

****

**Database Designer Salary (USA)**

How much does a Database Designer make?

The average Database Designer in the US makes $105,960. The average bonus for a Database Designer is $2,210 which represents 2% of their salary, with 100% of people reporting that they receive a bonus each year. Database Designers make the most in San Francisco at $158,132, averaging total compensation 49% greater than the US average.

|  |  |  |  |
| --- | --- | --- | --- |
| U.S. Average  $105,960  $19,969$537,961 | Base Salary  $103,750 | Bonus  $2,210 | Get Bonus  100% |

🡹 49% San Francisco 🡹 7% Seattle 🡹 5% New York

🡹3% Atlanta 🡻7% Austin 🡻13% Salt Lake City

**Salary Ranges for Database Designers**

The salaries of Database Designers in the US range from $19,969 to $537,961 , with a median salary of $96,081 . The middle 57% of Database Designers makes between $96,082 and $241,725, with the top 86% making $537,961.

**How much tax will you have to pay as a Database Designer**

For an individual filer in this tax bracket, you would have an estimated average federal tax in 2018 of 24%. After a federal tax rate of 24% has been taken out, Database Designers could expect to have a take-home pay of $86,240/year, with each paycheck equaling approximately $3,593\*.

**Quality of Life for Database Designer**

With a take-home pay of roughly $7,187/month, and the median 2BR apartment rental price of $2,506/mo\*\*, a Database Designer would pay 34.87% of their monthly take-home salary towards rent.

**Featured companies where database designers work**

Google

Microsoft

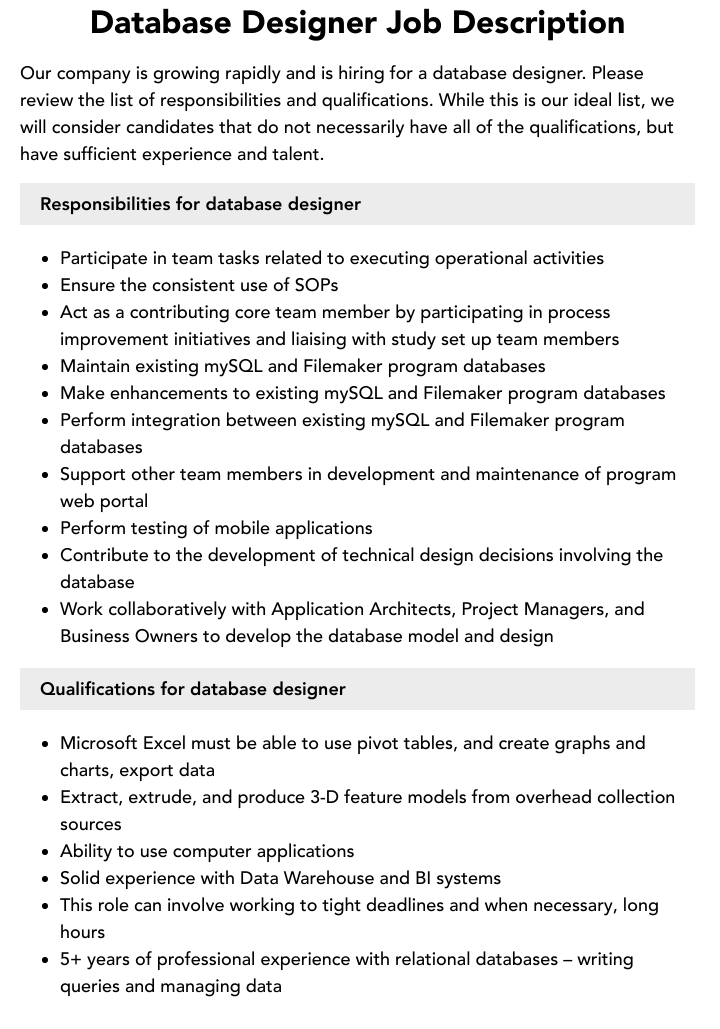
Linkedln

Amazon

Hubspot

Meta

****

**Application Designer**

The Application Designer is responsible to understand the business objectives and requirements and design critical components of an application to deliver the requested functionality and help achieve the client's business objectives.

** **

**Application Designers Salary**

**How much does an application designer earn?**

How much does an Application Designer make? The average Application Designer salary is $106,623 as of May 25, 2023, but the salary range typically falls between $94,995 and $119,367. Salary ranges can vary widely depending on many important factors, including education, certifications, additional skills, the number of years you have spent in your profession. With more online, real-time compensation data than any other website, helps you determine your exact pay target.

**Another method of earning up to 15-20 years**

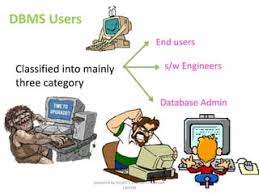
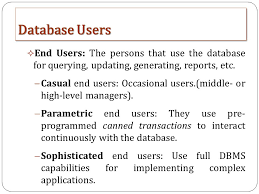
An Application Developer with 15 to 20 years of experience can earn an average of 1,450,700 LKR.

**Highest Paying Countries Of Application Designers**

|  |  |
| --- | --- |
| Country | Salary ($,£) |
| United States America | $106,120 (£73,246) |
| Switzerland | $101,449 (£70,022) |
| Denmark | $81,778 (£56,444) |
| Australia | $80,093 (£55,281) |
| Norway | $77,429 (£53,443) |
| Ireland | $76,747 (£52,272) |
| UK | $75,654 (£52,217) |
| Israel | $74,400 (£51,352) |
| New Zealand | $70,727 (£48,817) |
| Canada | $70,307 (£48,527) |

**End User**

End-user is a term used in product development that refers to the person who uses the product. An end-user database is, therefore, a database that is primarily used by a single person. A good example of this type of database is a spreadsheet stored on your local computer.

** **

End users are the people whose jobs require access to a database for querying, updating and generating reports.

**What DBMS apps are used by end users?**

* My SQL.
* Maria DB.
* Heidi SQL
* Microsoft SQL Server.
* Oracle DBMS.
* Postgre SQL.
* Mongo DB.
* Redis
* IBM DB2.





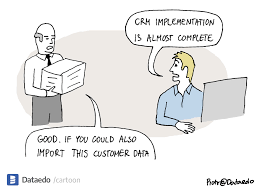
   

**Categories of End Users**

* Casual End users
* Naive or Parametric End Users
* Sophisticated End Users
* Specialized Users
* Application Programmers
* Standalone Users

**Casual End users**

These are the users who occasionally access the database but they require different information each time. They use a sophisticated database query language basically to specify their request and are typically middle or level managers or other occasional brow.



**Naive or Parametric End Users**

Parametric End Users are the unsophisticated who don't have any DBMS knowledge but they frequently use the database applications in their daily life to get the desired results. For examples, Railway's ticket booking users are naive users.

**Sophisticated End Users**

These users basically include engineers, scientists, business analytics, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their application to meet their complex requirements.

****

**Specialized Users**

Specialized users who write specialized database applications that do not fit into the fractional database processing framework. Application Programmer − The application programmer users who are responsible for developing the application programs or user interface.

****

**Application Programmers**

These users implement specific application programs to access the stored data. They must be familiar with the DBMSs to accomplish their task.

**Standalone Users**

These are those users whose job is basically to maintain personal databases by using a ready-made program package that provides easy-to-use menu-based or graphics-based interfaces, An example is the user of a tax package that basically stores a variety of personal financial data for tax purposes.

## THANK YOU!