

PROJECT MANAGEMENT AND ESTIMATION (PRO MASTI)

Project Management & Estimation



www.codeeverywhere.com

Email me your comments / suggestions

[\[madhu@madhu.iwarp.com\]](mailto:madhu@madhu.iwarp.com)

TABLE OF CONTENTS

- [1. ACKNOWLEDGEMENTS](#)
 - [2. ABSTRACT](#)
 - [3. INTRODUCTION TO RDBMS](#)
 - [4. ENTITY-RELATIONSHIP DIAGARM](#)
 - [5. RELATIONAL SCHEMA](#)
 - [6. IMPLEMENTATION DETAILS](#)
 - [7. INTRODUCTION TO ORACLE](#)
 - [8. INTRODUCTION TO SQL](#)
 - [9. INTRODUCTION TO DEVELOPER 2000](#)
 - [10. SQL SCRIPTS](#)
 - [11. SCREEN SHOTS](#)
 - [12. BIBLIOGRAPHIC REFERENCE](#)
-

ACKNOWLEDGEMENTS

[top](#)

We wish to express our sincere thanks to Mr. Raghavendra Bharadwaj, of ANZ Information Technology for his invaluable suggestions and for the time he spared for us in reviewing this project. We are also grateful to Mr.Inayath Khan ,Head of Dept of Computer Science,S.R.S.I.T for all his support and cooperation.

Finally we would like to thank all our friends and class mates for their countless suggestions towards improving the overall look of this project.

ABSTRACT

[top](#)

Software requirements specification:

The Existing System:

Project Managers monitoring the projects, and its status at any given time and taking into account the number of hours worked on a particular project by a particular employee. We try to ease the project manager's job by automating the system.

The Automated System:

The key features required of the product are:

1. The provision of an easy to use Graphical User Interface with a complete help system.
2. It should have some basic administration facility.
3. It should have the capability to add/remove/update employees and their skills submitted by user.
4. It should be capable of producing estimates for any given combination of project attributes so that the project manager can have a good idea of different possible estimates for the same project.

INTRODUCTION TO RDBMS

[top](#)

A Relational Database Management System (RDBMS) is a collection of programs that enables users to create and maintain a database. The characteristic that differentiates a DBMS from an RDBMS is that the RDBMS provides a set-oriented database language.

Some of the features of a RDBMS are:

1. Data is organized in terms of rows and columns in a table known as relations
2. The position of a row in a table is of no importance
3. The intersection of a row and a column must give a single value
4. All values appearing in the columns are derived from the underlying domain
5. Column names must be unique
6. All column values are atomic
7. In a relational database, there are no hard-coded relationships defined between tables. A relationship can be specified at any time using any column name.
8. Does not require the user to understand its physical implementation
9. Provides information about its content and structure in the system table

Dr. Codd's 12 Rules for a Relational Database Model

The most popular data storage model is the relational database, which grew from the seminal paper "A Relational Model of Data for Large Shared Data Banks," written by Dr. E. F. Codd in 1970. SQL evolved to service the concepts of the relational database model. Dr. Codd defined 13 rules, oddly enough referred to as Codd's 12 Rules, for the relational model:

0. A relational DBMS must be able to manage databases entirely through its relational capabilities.
1. Information rule-- All information in a relational database (including table and column names) is represented explicitly as values in tables.
2. Guaranteed access--Every value in a relational database is guaranteed to be accessible by using a combination of the table name, primary key value, and column name.
3. Systematic null value support--The DBMS provides systematic support for the treatment of null values (unknown or inapplicable data), distinct from default values, and independent of any domain.
4. Active, online relational catalog--The description of the database and its contents is represented at the logical level as tables and can therefore be queried using the database language.
5. Comprehensive data sub language--At least one supported language must have a well-defined syntax and be comprehensive. It must support data definition, manipulation, integrity rules, authorization, and transactions.
6. View updating rule--All views that are theoretically updateable can be updated through the system.
7. Set-level insertion, update, and deletion--The DBMS supports not only set-level retrievals but also set-level inserts, updates, and deletes.
8. Physical data independence--Application programs and ad hoc programs are logically unaffected when physical access methods or storage structures are altered.

9. Logical data independence--Application programs and ad hoc programs are logically unaffected, to the extent possible, when changes are made to the table structures.
10. Integrity independence--The database language must be capable of defining integrity rules. They must be stored in the online catalog, and they cannot be bypassed.
11. Distribution independence--Application programs and ad hoc requests are logically unaffected when data is first distributed or when it is redistributed.
12. Nonsubversion--It must not be possible to bypass the integrity rules defined through the database language by using lower-level languages.

Codd's idea for an RDBMS uses the mathematical concepts of relational algebra to break down data into sets and related common subsets. Because information can naturally be grouped into distinct sets, Dr. Codd organized his database system around this concept. Under the relational model, data is separated into sets that resemble a table structure. This table structure consists of individual data elements called columns or fields. A single set of a group of fields is known as a record or row.

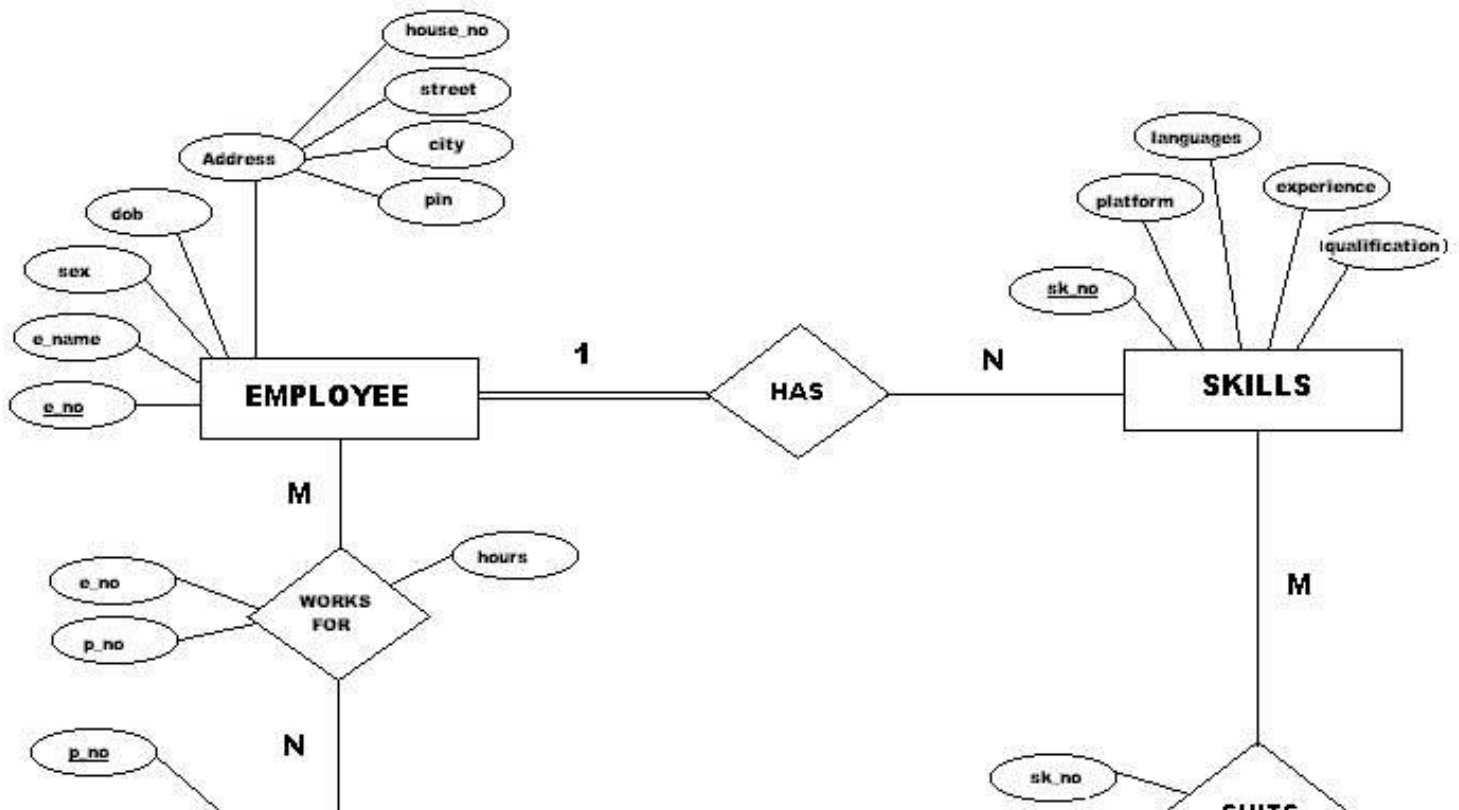
A modern RDBMS can perform a wide array of tasks, some of which are outlined below:

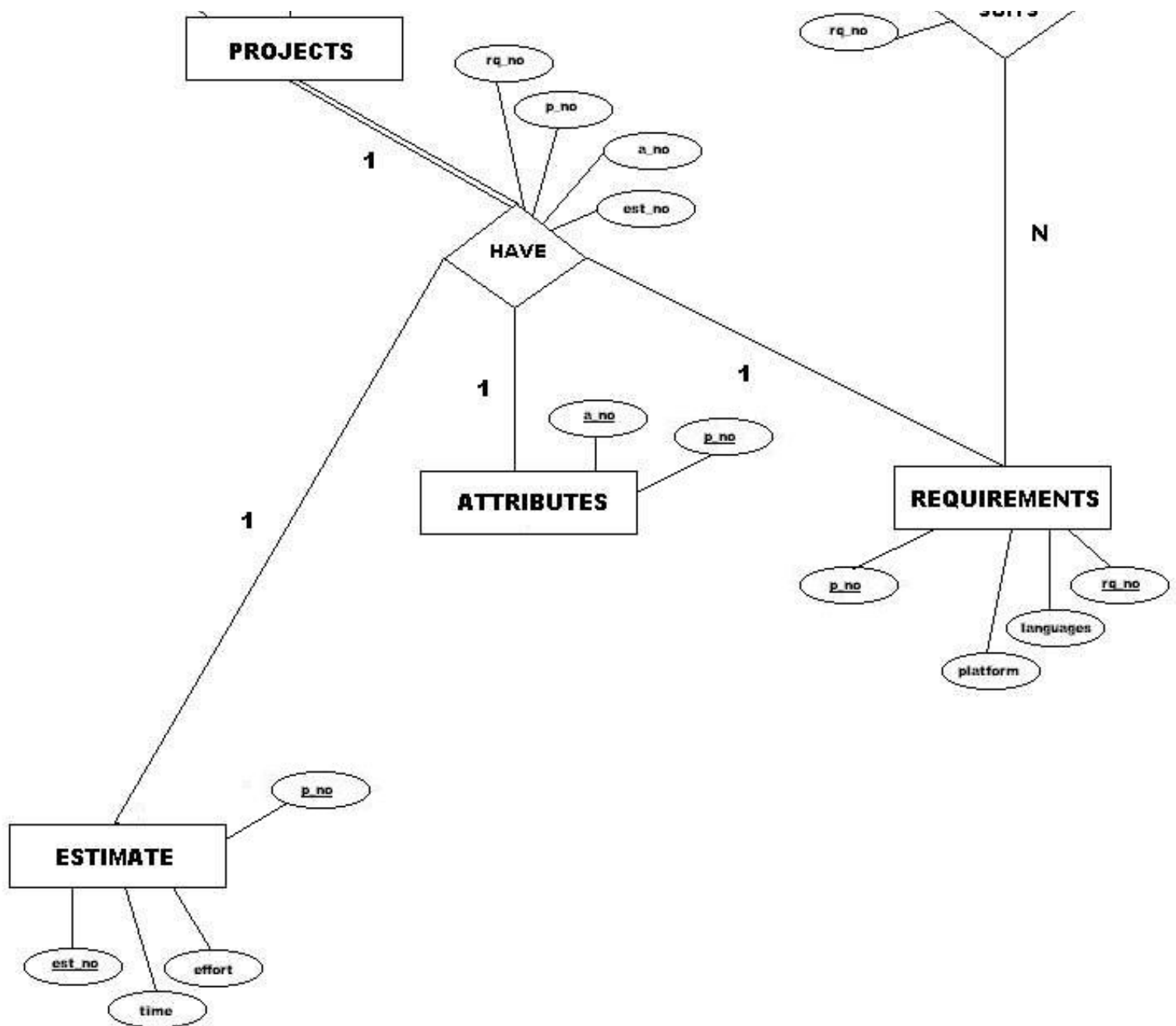
1. Define a database
2. Query the database
3. Provide access to the database
4. Modify the structure of the database
5. Provide documentation facilities
6. Secure data from unauthorized access
7. Communicate with networks
8. Export and import data

ENTITY RELATIONSHIP DIAGRAM

[top](#)

ER Diagram for *Project Management & Estimation*





Entities Used

- The **EMPLOYEE** entity maintains the personal details of every employee..
- The **SKILLS** table is used to store the employee skills, this will be used by the project manager to deploy employees on a specific project requiring people with some specific skillset.
- The **PROJECTS** table holds some general information about every project.
- The **ATTRIBUTE** table contains the 15 cost driver factors or project attributes which will be used to estimate the cost and time of the project using the COCOMO Model.
- The **REQUIREMENTS** table gives the requirements of each project in terms of human skill sets. The Project Manager can go through the Project Requirements report and Employee Skill report and allocate employees to projects.

ER SCHEMA

[top](#)

EMPLOYEE					
<u>E_NO</u>	NAME	SEX	DOB		
WORKS_ON					
<u>E_NO</u>	<u>P_NO</u>	HOURS			
SKILLS					
<u>SK_NO</u>	PLATFORM	LANGUAGES	EXPERIENCE	QUALIFICATION	
PROJECTS					
<u>P_NO</u>	P_NAME				
ATTRIBUTES					
<u>A_NO</u>	<u>P_NO</u>				
ESTIMATE					
<u>EST_NO</u>	TIME	EFFORT	<u>P_NO</u>		
REQUIREMENTS					
<u>RQ_NO</u>	PLATFORM	LANGUAGES	<u>P_NO</u>		

IMPLEMENTATION DETAILS

[top](#)

Front End	Developer 2000 / Oracle Forms 6i
Back End	Personal Oracle8
Report Generation	Oracle Reports 6i
Operating System	Windows 9x/Millennium Edition

Architecture:

1. Front-end Application

The user-friendly GUI that interact with the user is provided by Developer-2000. This an Oracle product found to be very efficient with Oracle as the backend since its component are geared towards rapid Application Development (RAD) and productivity. The three main components used are Forms, Reports and Graphics. Oracle Forms is a powerful 4GL for creating interactive applications. Oracle Reports allows a developer to generate reports which provide value addition to the application helping in the further analysis and growth.

2. Oracle Database

The Oracle database server contains the physical structures like tables, views, domains etc. that constitutes the database. Oracle employs the relational model for database management and is one of the most efficient RDBMS packages available. User requests are mapped on to the Oracle database using the Structured Query Language (SQL) along with its Procedural extensions (PL).

INTRODUCTION TO ORACLE

[top](#)

In 1977, Larry Ellison, Bob Miner, and Ed Oates formed a company called Relational Software Incorporated (RSI). This company built an RDBMS called Oracle. Ellison, Miner, and Oates made a key decision: to develop their RDBMS using C and the SQL interface. Soon after, they came out with version 1, a prototype. In 1979, RSI delivered its first product to customers. The Oracle RDBMS version 2 worked on the Digital PDP-11 running the RSX-11 operating system and was soon ported to the DEC VAX system.

1983 heralded the release of version 3, which touted changes in the SQL language as well as performance enhancements and other improvements. Unlike earlier versions, version 3 was written almost entirely in C. At this point, RSI changed its name to Oracle Corporation.

Oracle version 4 was released in 1984. This version supported both the VAX system and the IBM VM operating system. Version 4 was the first version to incorporate read consistency. Version 5, introduced in 1985, was a milestone because it introduced client/server computing to the market with the use of SQL*Net. Version 5 was also the first MS-DOS product to break through the 640KB barrier.

In 1988, Oracle presented version 6, which introduced low-level locking as well as a variety of performance improvements and functionality enhancements, including sequence generation and deferred writes. I was introduced to Oracle6 back in the days when we ran the TP1, TPC-A, and TPC-B benchmarks. At this point, Oracle was running on a large variety of different platforms and operating systems. In 1991, Oracle introduced the Oracle Parallel Server option on version 6.1 of the Oracle RDBMS on the DEC VAX platform. Soon the Parallel Server option was available on a variety of platforms.

Oracle7, released in 1992, included many architectural changes in the area of memory, CPU, and I/O utilization. Oracle7 is the full-featured RDBMS to which you are accustomed, the one you've been using for many years. Oracle7 introduced many advances in the area of ease of use, such as the SQL*DBA tools and database roles.

Finally, in 1997 Oracle introduced Oracle8, which added object extensions as well as a host of new features and administrative tools

The Oracle Architecture

The Oracle Database

The Oracle database has a logical layer and a physical layer. The physical layer consists of the files that reside on the disk; the components of the logical layer map the data to these physical components.

The Physical Layer

The physical layer of the database consists of three types of files:

- One or more datafiles--Datafiles store the information contained in the database. You can have as few as one datafile or as many as hundreds of datafiles. The information for a single table can span many datafiles or many tables can share a set of datafiles. Spreading tablespaces over many datafiles can have a significant positive effect on performance. The number of datafiles that can be configured is limited by the Oracle parameter MAXDATAFILES.
- Two or more redo log files--Redo log files hold information used for recovery in the event of a system failure. Redo log files, known as the redo log, store a log of all changes made to the database. This information is used in the event of a system failure to reapply changes that have been made and committed but that might not have been made to the datafiles. The redo log files must perform well and be protected against hardware failures (through software or hardware fault tolerance). If redo log information is lost, you cannot recover the system.

- One or more control files--Control files contain information used to start an instance, such as the location of datafiles and redo log files; Oracle needs this information to start the database instance. Control files must be protected. Oracle provides a mechanism for storing multiple copies of control files.

The Logical Layer

The logical layer of the database consists of the following elements:

- One or more tablespaces.
- The database schema, which consists of items such as tables, clusters, indexes, views, stored procedures, database triggers, sequences, and so on.

Conceptually Oracle is a kernel package and has a number of other supporting products that can be integrated with the kernel. Some of the common tools that can be integrated with the Oracle kernel are:

1. ORACLE Menu
2. ORACLE Form Builder
3. ORACLE Report Designer
4. Pre-compilers used for writing embedded SQL code

INTRODUCTION TO SQL

[top](#)

A Brief History of SQL

The history of SQL begins in an IBM laboratory in San Jose, California, where SQL was developed in the late 1970s. The initials stand for Structured Query Language, and the language itself is often referred to as "sequel." It was originally developed for IBM's DB2 product (a relational database management system, or RDBMS, that can still be bought today for various platforms and environments). In fact, SQL makes an RDBMS possible. SQL is a nonprocedural language, in contrast to the procedural or third-generation languages (3GLs) such as COBOL and C that had been created up to that time.

An Overview of SQL

SQL is the de facto standard language used to manipulate and retrieve data from these relational databases. SQL enables a programmer or database administrator to do the following:

- Modify a database's structure
- Change system security settings
- Add user permissions on databases or tables
- Query a database for information
- Update the contents of a database

SQL in Application Programming

SQL was originally made an ANSI standard in 1986. The ANSI 1989 standard (often called SQL-89) defines three types of interfacing to SQL within an application program:

- Module Language-- Uses procedures within programs. These procedures can be called by the application program and can return values to the program via parameter passing.
- Embedded SQL--Uses SQL statements embedded with actual program code. This method often requires the use of a precompiler to process the SQL statements. The standard defines statements for Pascal, FORTRAN, COBOL, and PL/1.
- Direct Invocation--Left up to the implementer.

Before the concept of dynamic SQL evolved, embedded SQL was the most popular way to use SQL within a program. Embedded SQL, which is still used, uses static SQL--meaning that the SQL statement is compiled into the application and cannot be changed at runtime. The principle is much the same as a compiler versus an interpreter. The performance for this type of SQL is good; however, it is not flexible--and cannot always meet the needs of today's changing business environments.

The ANSI 1992 standard (SQL-92) extended the language and became an international standard. It defines three levels of SQL compliance: entry, intermediate, and full. The new features introduced include the following:

- Connections to databases
- Scrollable cursors
- Dynamic SQL
- Outer joins

Dynamic SQL allows you to prepare the SQL statement at runtime. Although the performance for this type of SQL is not as good as that of embedded SQL, it provides the application developer (and user) with a great degree of flexibility. A call-level interface, such as ODBC or Sybase's DB-Library, is an example of dynamic SQL.

Call-level interfaces should not be a new concept to application programmers. When using ODBC, for instance, you simply fill a variable with your SQL statement and call the function to send the SQL statement to the database. Errors or results can be returned to the program through the use of other function calls designed for those purposes. Results are returned through a process known as the binding of variables.

INTRODUCTION TO DEVELOPER 2000

[top](#)

Developer 2000 is Oracle's productive Rapid Application Development (RAD) environment for building highly scalable, enterprise-class Internet database applications. It uses powerful declarative features so that business developers can instantly create fully functional applications from database definitions. It enables you to:

- Deploy applications on the Internet.
- Produce efficient, scalable applications.
- Build large, consistent, maintainable applications quickly.
- Use third-party tools and database servers.

Forms Developer Tools

Forms Developer provides a set of integrated builders that enable you to construct fully functional database applications with minimal effort. The main builder in Forms Developer is:

Form Builder

Form Builder is a powerful development tool for building robust, enterprise-class applications that enable end users to retrieve, enter, modify, and save information in the database. The highest-level Form Builder objects are modules. Each module consists of any number of lower-level objects.

When you work with Form Builder, you work with four types of modules, combining them to create a complete application.

- Form modules
- Menu modules
- Object library modules
- PL/SQL library modules

Blocks provide a mechanism for grouping related items into a functional unit. Just as database tables consist of related columns and rows, blocks contain related items that allow end users to store, display, and manipulate data records. There are two types of blocks:

- Data blocks

- Control blocks

A data block is associated with - or bound to - a database table or view, or a set of stored procedures. Most often, data blocks are based on a single database table.

A control block has no association with a database table or view; its objects do not relate to database columns.

Forms Developer includes additional tools to automate and manage application development tasks.

- Graphics Builder
- Project Builder
- Procedure Builder
- Schema Builder
- Query Builder
- Translation Builder

Reports Developer Tools

Reports Developer is a powerful enterprise reporting tool used to build reports that dynamically retrieve, format, and distribute information stored in the database. Reports Developer provides a builder that enables you to easily construct sophisticated reports and a server that enables you to securely publish reports to your users.

- Report Builder
- Reports Server

SQL Code

[top](#)

```
create table employee(
e_no number,
sk_no number,
f_name varchar2(20) not null,
l_name varchar2(20),
address varchar2(50),
phone_no varchar2(10),
email varchar2(20),
dob date,
sex char,
title varchar2(20)
);

create table skills(
sk_no number,
e_no number,
languages varchar2(30),
tools varchar2(30),
platform varchar2(30),
qualification varchar2(30) not null,
experience varchar2(30)
);

create table projects(
p_no number,
p_name varchar2(20) not null,
```

```
persondays number,  
no_of_persons number  
);
```

```
create table status(  
st_no number,  
phase varchar2(20),  
startdate date,  
deadline date,  
extension date  
);
```

```
create table attributes(  
a_no number,  
rely number(5,2),  
data number(5,2),  
cplx number(5,2),  
time number(5,2),  
stor number(5,2),  
virt number(5,2),  
turn number(5,2),  
acap number(5,2),  
aexp number(5,2),  
pcap number(5,2),  
lexp number(5,2),  
modp number(5,2),  
tool number(5,2),  
sced number(5,2),  
kdloc number(10),  
cons_a number(5,2),  
cons_b number(5,2)  
,p_no number);
```

```
create table estimate(  
est_no number,  
effort number(9,2),  
time number(9,2),  
p_no number  
);
```

```
create table requirements(  
rq_no number,  
languages varchar2(30),  
tools varchar2(30),  
platform varchar2(30),  
qualification varchar2(30),  
experience varchar2(30),  
p_no number  
);
```

```
create table works_on(  
e_no number,  
p_no number,
```

```
hours number(9,2)
);
```

```
create table password(
name varchar2(8),
password varchar2(8)
);
```

```
insert into password values('employee','employee');
insert into password values('manager','manager');
```

```
alter table employee
add constraint employeepk
primary key(e_no);
```

```
alter table skills
add constraint skillspk
primary key(sk_no);
```

```
alter table requirements
add constraint requirementspk
primary key(rq_no);
```

```
alter table status
add constraint statuspk
primary key(st_no);
```

```
alter table projects
add constraint projectspk
primary key(p_no);
```

```
alter table attributes
add constraint attributespk
primary key(a_no);
```

```
alter table estimate
add constraint estimatepk
primary key(est_no);
```

```
alter table works_on
add constraint works_onpk
primary key(p_no,e_no);
```

```
alter table employee
add constraint employeefk
```

```
foreign key(sk_no) references skills(sk_no)
on delete cascade;
```

```
alter table skills
add constraint skillsfk
foreign key(e_no) references employee(e_no)
on delete cascade;
```

```
alter table works_on
add constraint works_onfk1
foreign key(p_no) references projects(p_no)
on delete cascade;
```

```
alter table works_on
add constraint works_onfk2
foreign key(e_no) references employee(e_no)
on delete cascade;
```

```
alter table attributes
add constraint attributesfk
foreign key(p_no) references projects(p_no)
on delete cascade;
```

```
alter table estimate
add constraint estimatefk
foreign key(p_no) references projects(p_no)
on delete cascade;
```

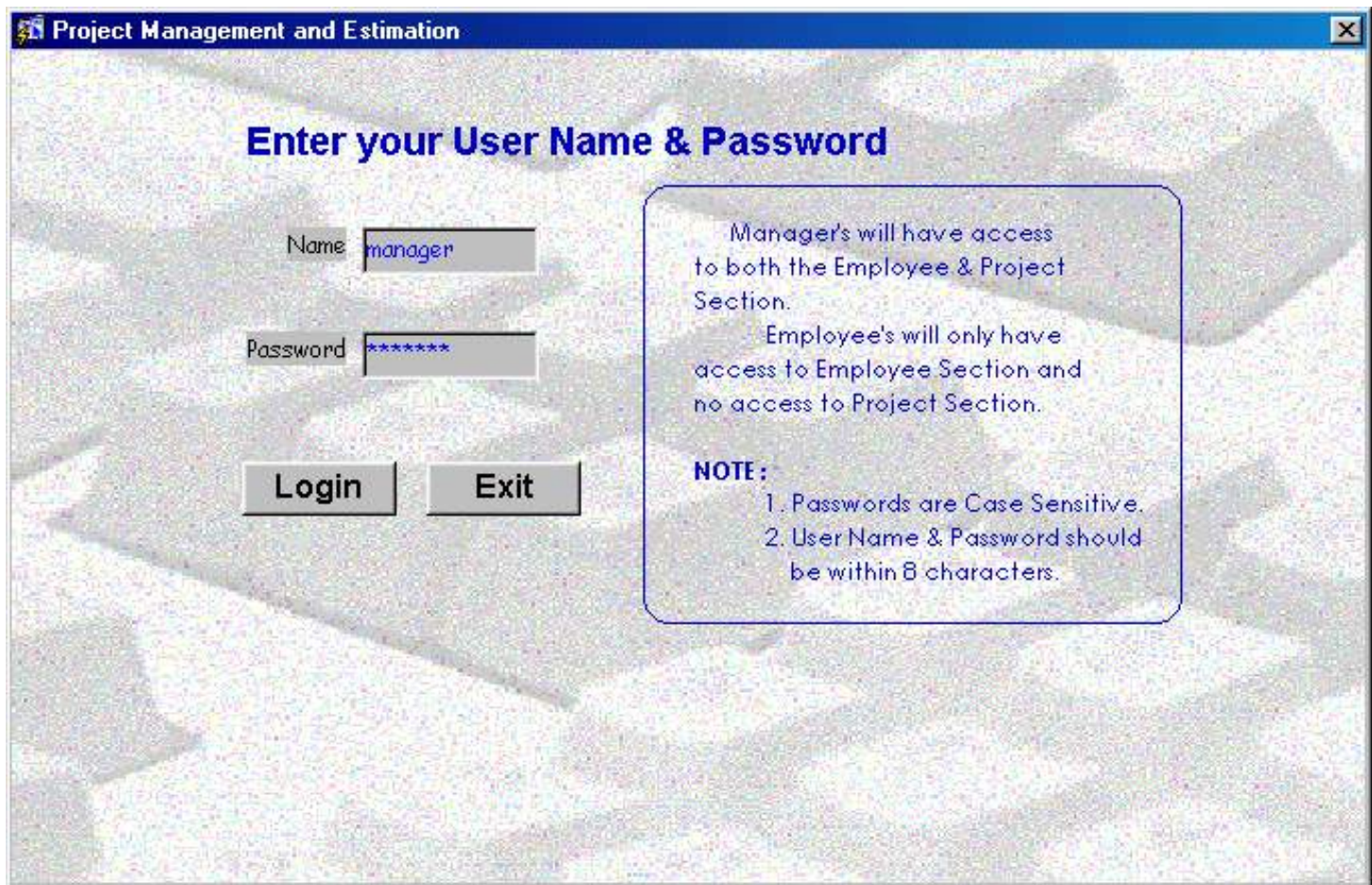
```
alter table requirements
add constraint requirementsfk
foreign key(p_no) references projects(p_no)
on delete cascade;
```

```
create view emp_view (Employee_No,First_Name,Title)
as select e_no,f_name,title
   from employee
   order by e_no;
```


```
create view prj_dtls
as select projects.p_no,projects.p_name,projects.no_of_persons,attributes.kdloc,
works_on.hours,estimate.effort
from projects,attributes,works_on,estimate
where projects.p_no=works_on.p_no
and projects.p_no=attributes.p_no
and projects.p_no=estimate.p_no;
```

SCREEN SHOTS

[top](#)



Project Management and Estimation



Employee Section

Project Section

Project Management & Estimation will give the project manager the ability to manage his staff and monitor many projects at a given time.

Also the project manager can do the cost and time estimations for his projects with **COCOMO** (Constructive Cost Model) as the basis.

www.CodeEverywhere.com

Employee Section

REPORTS

Here you can find out about all the employees in the company and there skill sets or there specilization.

Also you can find out about all the projects a particular employee is working on and the number of hours he has worked on each one of them.

Project Section

Projects

Allocate

Project View

Employee Details

REPORTS

Estimation Report

Requirements Report

In this section the project manager can enter details about new projects, see the requirements of the projects and assign employees to projects based on there skill sets.

Also the manager can do an cost estimation of the project based on the Constructive Cost Model.

Back

Employee - Works Master / Detail Form

Employee

F Name:

E No:

L Name:

Address:

Phone No:

Email:

Dob:

Sex:

Title:

Works On

E No	P No	Hours
1	1	25
1	9	20
1	12	25
1	13	11
1	14	20

View

Print

Back

First

Next

Prior

Last

Project Section

Projects Data Entry

P No

P Name*

Client Name

No Of Persons

Add
Modify
Save
Delete

Note :
'*' indicates that the field must be entered compulsarily.

Back COCOMO Estimation
Requirements

First Next Prior Last View

Project Section

Allocate Projects

E No

P No

Hours

Select from the following Projects

Find

P No	P Name
5	Pharmaceuticals
6	Food World Autm
7	Car Automation
8	Railway Reservation
9	Adv. Examination Sd.

Project Section

Estimation Using Constructive Cost Model

Project Attributes

Rely Data Cplx
 Software Reliability Req Database Size Product Complexity

Time Stor Virt
 Execution Time Constraint Main Storage Constraint Virtual Machine Volatility

Turn Acap Aexp
 Computer Turnaround Time Analyst Capability Applications Experience

Pcap Lexp Modp
 Programmer Capability Programming Language Experience Modern Programming Practice

Tool Sced
 Use of Software Tools Required Development Schedule

P No Cons A Cons B Kdloc
 Delivered Lines of Code in Thousand

Back Estimate

work_done: Previewer

File View Help

Page: 1

Work Done Report

Employee No.	First Name	Project Name	No. of Hours Worked
1	Madhu	PM & E	25
	Madhu	Adv Encryption Std	20
	Madhu	Graphics Editor	25
	Madhu	Web Tutor	11
	Madhu	Insurence	20
Total:			101
2	Naveen	PM & E	10
	Total:		
3	Srivathsa	Cricket Board DB	30
	Srivathsa	Adv Encryption Std	20
	Srivathsa	Graphics Editor	12
Total:			62
4	Harsha	Hotel Management	25
	Total:		
5	Kamal	Hotel Management	14
	Total:		
6	Santosh	Cricket Board DB	23

Santosh	Insurence	15
Total:		38

[top](#)

BIBLIOGRAPHIC REFERENCE

1. Fundamental of Database Systems (Second edition) - Ramez Elmasri and Shamkant B. Navathe
2. SQL & Relational Data Bases - Soren Vang
3. Commercial Application Development Using Oracle Developer 2000 Forms 5.0 - Ivan Bayross
4. Oracle - The Complete Reference
5. Oracle 7 The Application Development Handbook 2nd Edition - Alexis Leon
6. Mastering Oracle 7 & Client / Server Computing - Steven M. Bobrowski

Email me your comments / suggestions

[\[madhu@madhu.iwarp.com\]](mailto:madhu@madhu.iwarp.com)

[Screen Editor](#)

[Graphics Editor](#)

[DBMS](#)

[Data Security with AES](#)

[\[CodeEverywhere.Com\]](#)