ORACLE AND MYSQL

A Practical Approach

ORACLE AND MYSQL

A Practical Approach

By
B. Mohamed Ibrahim
MCA., M.Phil., MBA.,
Software Technical Consultant

FIREWALL MEDIA
(An Imprint of Laxmi Publications Pvt. Ltd.)
BANGALORE • CHENNAI • COCHIN • GUWAHATI • HYDERABAD
JALANDHAR • KOLKATA • LUCKNOW • MUMBAI • RANCHI
NEW DELHI • BOSTON, USA

Published by :

FIREWALL MEDIA

(*An Imprint of Laxmi Publications Pvt. Ltd.*)
113, Golden House, Daryaganj,
New Delhi-110002

Phone : 011-43 53 25 00
Fax : 011-43 53 25 28

www.laxmipublications.com
info@laxmipublications.com

Price : ₹ **125.00** *Only.* *First Edition :* 2013

OFFICES

✆ **Bangalore**	080-26 75 69 30	✆ **Chennai**	044-24 34 47 26
✆ **Cochin**	0484-237 70 04, 405 13 03	✆ **Guwahati**	0361-251 36 69, 251 38 81
✆ **Hyderabad**	040-24 65 23 33	✆ **Jalandhar**	0181-222 12 72
✆ **Kolkata**	033-22 27 43 84	✆ **Lucknow**	0522-220 99 16
✆ **Mumbai**	022-24 91 54 15, 24 92 78 69	✆ **Ranchi**	0651-221 47 64

FOR-3082-125-ORACLE & MYSQL-IBR **C—**
Typeset at : Sukuvisa Enterprises, New Delhi. *Printed at* : Mehra Offset Press

CONTENTS

Chapters	*Pages*
1. Introduction to Database System	**1–29**
1.0 Introduction	1
1.1 RDBMS	1
1.2 Database Keys	2
1.3 Normalization	3
Summary	29
2. Working with Oracle Database	**30–84**
2.0 Introduction	30
2.1 Installing Oracle	30
2.2 Adding/Deleting a User	33
2.3 Executing SQL Commands	40
2.4 SQL	42
2.5 Backup and Restore Databases	66
2.6 PL/SQL	66
Summary	84
3. Working with MySQL Database	**85–133**
3.0 Introduction	85
3.1 Installing MySQL	85
3.2 Creating/Dropping a Database Schema	86
3.3 Creating/Dropping a User	87
3.4 Data Types	88
3.5 SQL Commands	90
3.6 MySQL Storage Engines	94
3.7 MySQL Programs	95
3.8 Built-in Functions	97
3.9 PL/SQL	100
3.10 Illustrating Examples	104
Summary	133
Index	**134–136**

PREFACE

The Database Management System (DBMS) is a software product that organises the storage and retrieval of data with database. Major developments are going on in the DBMS field to achieve the required security, integrity of data and easy to work with it. Those developments include: RDBMS (Relational Database Management System), OODMS (Object Oriented Database Management System), and ORDBMS (Object RDBMS).

Oracle is the most widely used RDBMS, and MySQL is the most popular open-source RDBMS. Both of these databases provide support for database connectivity to high-level languages. This book focuses on these two database systems, with basics to intermediate level.

Chapter 1 (Introduction to Database Systems): This chapter introduces the basic concepts of databases. The features of Relational DBMS, various types of keys used in database design are explored. The Normalization process is explained in detail with the Normal Forms. Suitable example queries with their corresponding output are given for easy understanding of concepts.

Chapter 2 (Working with Oracle Database): This chapter is devoted for Oracle database. This chapter starts with Oracle Installation. The SQL commands are grouped as Data Definition Language (DDL), Data Manipulation Language (DML) and Transaction Control Language (TCL) commands and explained with example queries. The database objects such as views, synonyms, and sequences are explained with their general formats. The PL/SQL for Oracle is explained in detail with sample code. The stored procedures are neatly explained, which can be called by an application through database connectivity to minimise database interactions in performing bulk database operations. Some basic database administrative operations such as creating databases, creating users, back-up and restore databases, grant and revoke privileges on database are given.

Chapter 3 (Working with MySQL Database): This chapter explains the important concepts to work with MySQL database. This chapter started with the MySQL installation procedure for Microsoft Windows platform. Important data-definition, data-manipulation, transaction-control, and MySQL programs are explained with illustrations. A set of frequently used built-in functions are given by their operations wise. The latest features of MySQL, stored procedures / functions, triggers, are explained with example code.

This book is intensely written for the beginners and intermediates in database knowledge, and those are using databases in high-level languages for developing applications. It is written in a practical approach that the readers will find suitable example code with output wherever they expect them.

I thank the Almighty Allah, who gave me the soul, strength, knowledge and everything. All praise is due to him. Salutations on the Prophet (may Allah's blessings and peace be upon him) and his descendents. I thank all the well wishers of mine, who prayed the Almighty for me.

—Author

Chapter 1

INTRODUCTION TO DATABASE SYSTEMS

1.0 INTRODUCTION

A datum represents a raw fact and figure, a meaningful datum gives information. A group of related data forms a record. A file is composed of several related records. A database is an integrated collection of related files. The Database Management System (DBMS) is a software product that organises the storage and retrieval of data with database. Major developments are going on in the DBMS field to achieve the required security, integrity of data and easy to work with it. Those developments include: RDBMS (Relational Database Management System), OODMS (Object Oriented Database Management System), and ORDBMS (Object RDBMS). This chapter briefs the basic database concepts.

1.1 RDBMS

The Database Management System (DBMS) is a concept to store, access, and manipulate the data in an effective way. Generally the data are stored in the format of tables (records), comprised of rows (tuple) and columns (attributes); these tables form a database. To make relational (compare and combine) operations on database tables, the Relational Database Management System (RDBMS) is designed. Popular RDBMS software include *Oracle*, *MySQL*, *MS SQL Server*, *Sybase*, and *Informix*. Every RDBMS software should satisfy the following rules, defined by Boyce and Codd (also called as Boyce-Codd rules). However, conflicts are still there that some of the RDBMS software do not satisfy all these twelve rules.

Boyce-codd rules:

1. The information rule
2. The rule of guaranteed access

3. The systematic treatment of all *null* values
4. The database description rule
5. The comprehensive data sub language rule
6. The view updating rule
7. The insert and update rule
8. The physical data independence rule
9. The logical data independence rule
10. The integrity independence rule
11. The distribution rule
12. No sub version rule.

1.2 DATABASE KEYS

The database keys are used to keep the database records straight, and forms structure of database tables. There are many keys defined in database, some important and frequently referred keys are briefed as below.

Key: A key is a single or combination of attributes in a database table.

Primary Key: An attribute or a combination of attributes that uniquely identifies a record in a database table.

Unique Key: It a constraint that is applied on an attribute or a combination of attributes for uniquely identifying the values in that column(s), but the column(s) can accept NULL values.

Difference between Primary Key and Unique Key: Primary key will not accept NULL values, however Unique key will accept.

Composite Key/Concatenate Key: A primary key that made up of more than one attributes.

Foreign Key: A primary key can be references to another table as a Foreign Key. An attribute or a combination of attributes in a table which matches a primary key in another table.

Composite Primary Key: A Primary key consists of a combination of attributes (more than one column).

Super Key: A Super Key is a combination of attributes that can be uniquely used to identify a database record. A table may have many Super Keys.

Candidate Key: A candidate key is an attribute/a combination of attributes that can be uniquely used to identify a database without any extraneous information. A Candidate Keys are a sub-set of Super Keys that do not have any extraneous information in them. Each table may have one or more candidate keys. Primary key for the table is selected from these candidate keys. Any of the candidate key that is not part of the primary key is called Alternate Key.

Surrogate Key: It is just a unique value in each record that uniquely identifies the record and act as a primary key. The only requirement for a surrogate primary key is that it is unique for each row in the table. These Surrogate keys are not derived from application data; it is generally generated by RDBMS itself such as Sequences (Oracle), Auto Increment (MySQL), UUIDs (Universal Unique Identifiers), and GUIDs (Globally Unique Identifiers).

1.3 NORMALIZATION

"Normalization is the optimisation process of efficiently organizing data in a database so that redundancy of data will be eliminated, data dependencies will be ensured and data confirms to the defined integrity constraints."

Data dependency includes grouping only strongly related data in a table (for example, in employee table, one can store employee_number, employee_name, salary, etc., but not project_duration. The integrity constraints ensure the adherence of defined rules on data, for example, the mark column in marklist table should accept value within the range 0-100.

The Readers who are new to database may find difficulties in understanding the SQL statements used in the examples with the following sections. Please refer forth-coming chapters for detailed explanation of SQL statements.

1.3.1 Normal Forms

They are the rules and guidelines to make ensure a database is normalized.

1.3.1.1 First Normal Form (1NF)

"A database table is in the first normal form if it contains only atomic values."

Focus: Eliminate repeating groups/group value in fields.

Atomic values refer no repeating groups (set of columns of same kind of data, for example, email1, email2 … and address1, address2 …), no group of fields (group of filed: the value with divisible parts, for example, address column contains a group value which can be divisible to door_number, street_name, city_name, country_name, postal_code …).

Example:

Note: I am using SQL Client command-line interface for executing SQL statements.

```
SQL> create table employee (
     empnumber number(5) primary key,
     empname   varchar2(30),
     salary    number(9,2),
     emails    varchar2(100)
     );
Table created.
SQL> insert into employee (empnumber, empname, salary, emails) values
     (1001, 'Mohamed Ibrahim', 92000.0,
     'bmohamedibrahim@yahoo.co.in, bmdibrahim@rediffmail.com');
1 row created.
SQL> select * from employee;
EMPNUMBER EMPNAME          SALARY EMAILS
---- --------------- ----- -------------------------------------------------------
1001 Mohamed Ibrahim 92000 bmohamedibrahim@yahoo.co.in, bmdibrahim@rediffmail.com
```

The above employee table contains non-atomic value for the columns emails and empname. The emails filed contains many email addresses (in our example, two emails) separated by comma.

The empname filed contains both first name and last name. We can change the design of the above table as below:

```
SQL> drop table employee;
Table dropped.

SQL> create table employee
     ( empnumber    number(4) primary key,
     empfirstname   varchar2(16),
     emplastname    varchar2(16),
     salary         number(9,2),
     email1         varchar2(50),
     email2         varchar2(50)
     );
Table created.

SQL> insert into employee (empnumber, empfirstname, emplastname,
     salary, email1, email2) values
     (1001, 'Mohamed', 'Ibrahim', 92000.0,
     'bmohamedibrahim@yahoo.co.in', 'bmdibrahim@rediffmail.com');
1 row created.
```

In the above table, even though no group of values present in columns, repeating groups occurred. The columns email1 and email2 are repeating groups. We can use a separate table to store email addresses and map them with our employee table using foreign key reference as below.

```
SQL> create table employee
     ( empnumber      number(4) primary key,
     empfirstname     varchar2(16),
     emplastname      varchar2(16),
     salary           number(9,2)
     );
Table created.

SQL> create table email_address
     ( id           number(5) primary key,
     email          varchar2(50),
     empnumber      number(5),
     foreign key    (empnumber) references employee(empnumber)
                                        on delete cascade
     );
Table created.

SQL> insert into employee (empnumber, empfirstname, emplastname,
     salary) values(1001, 'Mohamed', 'Ibrahim', 92000.0);
1 row created.
```

```
SQL> insert into email_address (id, email, empnumber) values
     (1, 'bmohamedibrahim@yahoo.co.in', 1001);
1 row created.

SQL> insert into email_address (id, email, empnumber) values
     (2, 'bmdibrahim@rediffmail.com', 1001);
1 row created.

SQL> select * from employee;
     EMPNUMBER   EMPFIRSTNAME  EMPLASTNAME     SALARY
     ----------  ------------  ------------    ----------
     1001        Mohamed       Ibrahim         92000

SQL> select * from email_address;
       ID EMAIL                        EMPNUMBER
------ ---------------------------     ----------
       1 bmohamedibrahim@yahoo.co.in   1001
       2 bmdibrahim@rediffmail.com     1001
```

1.3.1.2 Second Normal Form (2NF)

"A database table is in the second normal form if it fulfils all the requirements of the first normal form and all non-primary-key fields in the table must be fully dependent on the entire primary key of the table."

Focus: Eliminate redundant data on non-primary and non-foreign key fields.

The entire primary key refers the whole set of fields of composite primary key. The 2NF enforces no partial dependencies. The 2NF further forces to remove duplicate values in the columns of a table.

The composite primary key is a primary key in a table made up of more than one column. In the following example, we are defining the combination of firstname and lastname in employee table as composite primary key, so that duplicate values on either firstname or lastname are allowed, but a duplicate value on the combination of these two fields are not allowed.

```
SQL> drop table email_address;
Table dropped.

SQL> drop table employee;
Table dropped.

SQL> create table employee
     (firstname  varchar2(16),
     lastname    varchar2(16),
     salary      number(9,2),
```

```
     constraint  employee_pk primary key (firstname, lastname)
     );
Table created.

SQL> insert into employee (firstname, lastname, salary) values
     ('mohamed', 'ibrahim', 98000.0);
1 row created.

SQL> insert into employee (firstname, lastname, salary) values
     ('mohamed', 'yasin', 87000.0);
1 row created.

SQL> insert into employee (firstname, lastname, salary) values
     ('syed', 'ibrahim', 82500.0);
1 row created.

SQL> select * from employee;
     FIRSTNAME     LASTNAME      SALARY
     ----------    ---------     ----------
     mohamed       ibrahim       98000
     mohamed       yasin         87000
     syed          ibrahim       82500

SQL> insert into employee (firstname, lastname, salary) values
     ('syed', 'ibrahim', 50000.0);
insert into employee (firstname, lastname, salary) values
*
ERROR at line 1:
ORA-00001: unique constraint (IBRAHIM.EMPLOYEE_PK) violated
```

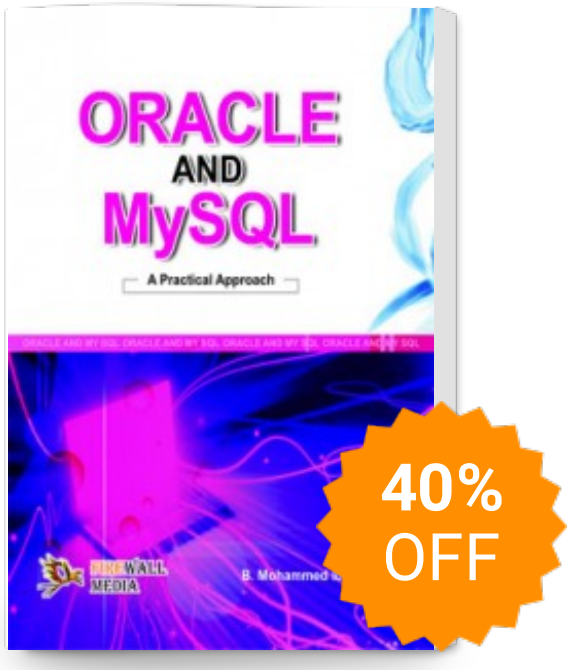
Example

Now, let us take one example to illustrate second normal form.

```
SQL> create table teacher_info
     ( teacher_id        number(4),
     teacher_first_name  varchar2(16),
     teacher_last_name   varchar2(16),
     class_id            varchar2(4),
     class_name          varchar2(40),
     constraint teacher_info_pk primary key (teacher_id, class_id)
     );
Table created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
```

```
     values (1001, 'Mohamed', 'Ibrahim', 'CS', 'Computer Science');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1002, 'Mohamed', 'Yaseen', 'CS', 'Computer Science');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1003, 'Karan', 'Singh', 'MGT', 'Management');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1003, 'Karan', 'Singh', 'COM', 'Commerce');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1004, 'Kiran', 'Kumar', 'COM', 'Commerce');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1005, 'Akhfa', 'Nafi', 'COM', 'Commerce');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1006, 'Ravi', 'Chandren', 'MGT', 'Management');
1 row created.

SQL> insert into teacher_info (teacher_id, teacher_first_name,
     teacher_last_name, class_id, class_name)
     values (1007, 'Nitin', 'Otageri', 'ZOO', 'Zoology');
1 row created.

SQL> select * from teacher_info;
     TEACHER_ID  TEACHER_FIRST_NA  TEACHER_LAST_NAM  CLAS  CLASS_NAME
     ----------  ----------------  ----------------  ----  ----------------
     1001        Mohamed           Ibrahim           CS    Computer Science
     1002        Mohamed           Yaseen            CS    Computer Science
```

Oracle And Mysql - A Practical Approach By B. Mohamed Ibrahim



Publisher : Laxmi Publications ISBN : 9789380298665

Author : B. Mohamed Ibrahim