Oracle SQL

Document Revision History

Date	Revision No.	Author	Summary of Changes
July 2016	1.0	Shraddha Jadhav	Initial Draft

Table of Contents

Getting Started

Overview

This lab book is a guided tour for learning Oracle SQL. It comprises 'To Do' assignments. Follow the steps provided and work out the 'To Do' assignments.

Setup Checklist for DBMS SQL

Here is what is expected on your machine in order for the lab to work.

Minimum System Requirements

- Intel Pentium 90 or higher (P166 recommended)
- Microsoft Windows 95, 98, or NT 4.0, 2k, XP.
- Memory: 32MB of RAM (64MB or more recommended)

Please ensure that the following is done:

- Oracle Client installed.
- Connectivity to Oracle Server

Instructions

- For all coding standards refer Appendix A. All lab assignments should refer coding standards.
- Create a directory by your name in drive <drive>. In this directory, create a subdirectory <DBMS SQL>_assgn. For each lab exercise create a directory as lab <lab number>.

Learning More (Bibliography if applicable)

NA

Problem Statement / Case Study

Table descriptions

Emp

Name	Null?	Type
Empno	Not Null	Number(4)
Ename	-	Varchar2(10)
job		Varchar2(9)
mgr		Number(4)
Hiredate		Date
Sal	-	Number(7,2)
Comm	-	Number(7,2)
Deptno	-	Number(2)

Designation_Masters

Name	Null?	Type
Design_code	Not Null	Number(3)
Design_name		Varchar2(50)

Department_Masters

Name	Null?	Type
Dept_Code	Not Null	Number(2)
Dept_name		Varchar2(50)

Student_Masters

Name	Null?	Type
Student_Code	Not Null	Number(6)
Student_name	Not Null	Varchar2(50)
Dept_Code		Number(2)
Student_dob		Date
Student_Address		Varchar2(240)

Student_Marks

Name	Null?	Type

Student_Code		Number(6)
Student_Year	Not Null	Number
Subject1		Number(3)
Subject2		Number(3)
Subject3		Number(3)

Staff_Masters

Name	Null?	Type
Staff_code	Not Null	Number(8)
Staff_Name	Not Null	Varchar2(50)
Design_code		Number
Dept_code		Number
HireDate		Date
Staff_dob		Date
Staff_address		Varchar2(240)
Mgr_code		Number(8)
Staff_sal		Number (10,2)

Book_Masters

Name	Null?	Type
Book_Code	Not Null	Number(10)
Book_Name	Not Null	Varchar2(50)
Book_pub_year		Number
Book_pub_author	Not Null	Varchar2(50)

Book_Transactions

Name	Null?	Type
Book_Code		Number
Student_code		Number
Staff_code		Number
Book_Issue_date	Not Null	Date
Book_expected_return_date	Not Null	Date
Book_actual_return_date		Date

Lab 1. Basic SQL Command

Goals	• Query the database
Time	2 hr 30 min

1. Retrieve the details (Name, Salary and dept code) of the staff who are working in department code 20, 30 and 40.
2. Display the code and total marks for every student. Total Marks will be calculated as subject1+subject2+subject3 .(Refer Student_marks table)
3. List the Name and Designation code of the staff who have joined before Jan 2003 and whose salary range is between 12000 and 25000. Display the columns with user defined Column headers. Hint: Use As clause along with other operators
4. List the code, name, and department number of the staff who have experience of 18 or more years and sort them based on their experience.
5. List the name, designation code, and salary for 10 years of the staff who are working in departments 10 and 30.
6. Display name concatenated with dept code separated by comma and space. Name the column as 'Student Info'.
7. Display the staff details who do not have manager. Hint: Use is null
8. Write a query which will display name, department code and date of birth of all students who were born between January 1, 1981 and March 31, 1983. Sort it based on date of birth (ascending).Hint: Use between operator
9. Display the Book details that were published during the period of 2001 to 2004. Also display book details with Book name having the character '&' anywhere.
10. Display the Book details where the records have the word "COMP" anywhere in the Book name.
11. List the details of the staff, whose names start with 'A' and end with 'S' or whose names contains N as the second or third character, and ending with either 'N' or 'S'.
12. List the names of the staff having '_' character in their name.

13. Create the Customer table with the following columns.

CustomerId	Number(5)
Cust_Name	varchar2(20)
Address1	Varchar2(30)
Address2	Varchar2(30)

a. **2.** Modify the Customer table Cust_Name column of datatype with Varchar2(30), rename the column to CustomerName and it should not accept Nulls.

14. a) Add the following Columns to the Customer table.

i.	Gender	Varchar2(1)
ii.	Age	Number(3)
iii.	PhoneNo	Number(10)

b) Rename the Customer table to Cust_Table

15. Insert rows with the following data in to the Customer table.

a. Insert into customer values: (1000, 'Allen', '#115 Chicago', '#115 Chicago', 'M', '25, 7878776')

b. In similar manner, add the below records to the Customer table:

   a. 1000, Allen, #115 Chicago, #115 Chicago, M, 25, 7878776
   b. 1001, George, #116 France, #116 France, M, 25, 434524
   c. 1002, Becker, #114 New York, #114 New York, M, 45, 431525

16. Insert the row given below in the Customer table and see the message generated by the Oracle server.

   a. 1002, John, #114 Chicago, #114 Chicago, M, 45, 439525

17. Delete all the existing rows from Customer table, and let the structure remain itself using TRUNCATE statement.

18. In the Customer table, add a column E_mail.

19. Drop the E_mail column from Customer table.

20. Add a new column EmailId to Customer table.

21. Mark EmailId column as unused before dropping it.

22. Drop the unused EmailId column from the Customer table.

23. Create the Suppliers table based on the structure of the Customer table. Include only the CustomerId, CustomerName, Address1, Address2, and phoneno columns.

24. Name the columns in the new table as SuppID, SName, Addr1, Addr2, and Contactno respectively.

25. Drop the above table and recreate the following table with the name CustomerMaster.

iv.	CustomerId	Number(5) Primary key(Name of constraint is CustId_PK)
v.	CustomerName	Varchar2(30) Not Null
vi.	Addressl	Varchar2(30) Not Null
vii.	Address2	Varchar2(30)
viii.	Gender	Varchar2(l)
ix.	Age	Number(3)
x.	PhoneNo	Number(10)

26. Create Employee table with same structure as EMP table.

SQL>Create table employee as select * from emp where 1=3

SQL>desc employee

Name	Null?	Type
EMPNO	NOT NULL	NUMBER(4)
ENAME		VARCHAR2(10)
JOB		VARCHAR2(50)
MGR		NUMBER(4)
HIREDATE		DATE
SAL		NUMBER(7,2)
COMM		NUMBER(7,2)
DEPTNO		NUMBER(2)

SQL>select * from employee

27. Write a query to populate Employee table using EMP table's empno, ename, sal, deptno columns.

SQL>select * from employee

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH				800		20
7499	ALLEN				1600		30
7521	WARD				1250		30
7566	JONES				2975		20
7654	MARTIN				1250		30

7698	BLAKE				2850		30
7782	CLARK				2450		10
7788	SCOTT				3000		20
7839	KING				5000		10
7844	TURNER				1500		30
7876	ADAMS				1100		20
7900	JAMES				950		30
7902	FORD				3000		20
7934	MILLER				1300		10

14 rows selected.

28. Write a query to change the job and deptno of employee whose empno is 7698 to the job and deptno of employee having empno 7788.

29. Delete the details of department whose department name is ‘SALES’.

30. Write a query to change the deptno of employee with empno 7788 to that of employee having empno 7698.

31. Insert the following rows to the Employee table through parameter substitution.
    a. 1000,Allen, Clerk,1001,12-jan-01, 3000, 2,10
    b. 1001,George, analyst, null, 08 Sep 92, 5000,0, 10
    c. 1002, Becker, Manager, 1000, 4 Nov 92, 2800,4, 20
    d. 1003, 'Bill', Clerk, 1002, 4 Nov 92,3000, 0, 20

32. Create a Project Table with below structure

Name	Null?	Type
PROJID	NOT NULL	VARCHAR2(10)
PROJ_NAME		VARCHAR2(25)
START_DATE		DATE
END_DATE		DATE

- Insert Records into Project Table
- Create Employee_Project Table that will have Empno and Project Id as Primary key. Insert records into Employee_Project Table using inline view

33. Insert rows with the following data into the Customer table. 6000, John, #115 Chicago, #115 Chicago, M, 25, 7878776, 10000

- 6001, Jack, #116 France, #116 France, M, 25, 434524, 20000
- 6002, James, #114 New York, #114 New York, M, 45, 431525, 15000.50

Use parameter substitution.

34. Create a Savepoint named ‘SP1’ after third record in the Customer table .

35. Insert the below row in the Customer table.

6003, John, #114 Chicago, #114 Chicago, M, 45, 439525, 19000.60

36. Execute rollback statement in such a way that whatever manipulations done before Savepoint sp1 are permanently implemented, and the ones after Savepoint SP1 are not stored as a part of the Customer table.

Lab 2. Operators, Single Row Functions

Goals	• Querying tables using single row functions • Querying tables using date functions • Querying tables using number functions • Querying tables using group functions
Time	2 hr 30 min

2.1: Single Row Functions:

1. Create a query which will display Staff Name, Salary of each staff. Format the salary to be 15 character long and left padded with '$'.

2. Display name and date of birth of students where date of birth must be displayed in the format similar to "January, 12 1981" for those who were born on Saturday or Sunday.

3. Display each Staff name and number of months they worked for the organization. Label the column as 'Months Worked'. Order your result by number of months employed. Also Round the number of months to closest whole number.

4. Display the name and salary of the staff. Salary should be represented as X. Each X represents a 1000 in salary. Hint: Divide salary by 1000, use rpad to substitute an 'X' for every 1000.

Sample Output

```
JOHN          10000          XXXXXXXXXX
ALLEN         12000          XXXXXXXXXXXX
```

5. List the details of the staff who have joined in first half of December month (irrespective of the year).

6. Write a query that displays Staff Name, Salary, and Grade of all staff. Grade depends on the following table.

Salary	Grade
Salary >=50000	A
Salary >= 25000 < 50000	B
Salary>=10000 < 25000	C
OTHERS	D

7. Display the Staff Name, Hire date and day of the week on which staff was hired. Label the column as DAY. Order the result by the day of the week starting with Monday.   Hint :Use to_char with hiredate and formats 'DY' and 'D'
8. Show staff names with the respective numbers of asterisk from Staff_Masters table except first and last characters. For example: KING will be replaced with K**G. .   Hint: Use substring, rpad and length functions.
9. Write a query to find the position of third occurrence of 'i' in the given word 'Mississippi'.
10. Write a query to find the pay date for the month. Pay date is the last Friday of the month. Display the date in the format "Twenty Eighth of January, 2002". Label the heading as PAY DATE. Hint: use to_char,next_day and last_day functions
11. Display Student code, Name and Dept Name. Display "Electricals" if dept code = 20, "Electronics" if Dept code =30 and "Others" for all other Dept codes in the Dept Name column. Hint : Use Decode
12. Display the student name and department code of students. If student does not belong to any department, display "No Department". Label the column as "Department". (Hint: Use NVL function)
13. Because of budget issues, the HR department needs a report that displays the last name and salary of employees who earn more than $12,000. Save your SQL statement as a file named lab_02_01.sql. Run your query.
14. Open a new SQL Worksheet. Create a report that displays the last name and department number for employee number 176. Run the query.
15. The HR department needs to find high-salary and low-salary employees. Modify lab_02_01.sql to display the last name and salary for any employee whose salary is not in the range of $5,000 to $12,000. Save your SQL statement as lab_02_03.sql.
16. Create a report to display the last name, job ID, and hire date for employees with the last names of Matos and Taylor. Order the query in ascending order by the hire date.
17. Display the last name and department ID of all employees in departments 20 or 50 in ascending alphabetical order by name.
18. Modify to display the last name and salary lab_02_03.sql of employees who earn between $5,000 and $12,000, and are in department 20 or 50. Label the columns Employee and Monthly Salary, respectively. Save lab_02_03.sql as lab_02_06.sql again. Run the statement in lab_02_06.sql.
19. The HR department needs a report that displays the last name and hires date for all employees who were hired in 1994.

20. Create a report to display the last name and job title of all employees who do not have a manager.

21. Create a report to display the last name, salary, and commission of all employees who earn commissions. Sort data in descending order of salary and commissions.
22. Use the column's numeric position in the ORDER BY clause.

23. Members of the HR department want to have more flexibility with the queries that you are writing. They would like a report that displays the last name and salary of employees who earn more than an amount that the user specifies after a prompt. Save this query to a file named. If you enter lab_02_10.sql 12000 when prompted, the report displays the following results:

24. The HR department wants to run reports based on a manager. Create a query that prompts the user for a manager ID and generates the employee ID, last name, salary,and department for that manager's employees. The HR department wants the ability to sort the report on a selected column. You can test the data with the following values:
    manager_id = 103, sorted by last_name:
    manager_id = 201, sorted by salary:
    manager_id = 124, sorted by employee_id:
    If you have time, complete the following exercises:

25. Display all employee last names in which the third letter of the name is "a."

26. Display the last names of all employees who have both an "a" and an "e" in their last name.

27. Display the last name, job, and salary for all employees whose jobs are either those of a sales representative or of a stock clerk, and whose salaries are not equal to $2,500, $3,500, or $7,000.

28. Modify to display the last name, salary, a lab_02_06.sql nd commission for all employees whose commission is 20%. Save lab_02_06.sql as lab_02_15.sql again. Rerun the statement in lab_02_15.sql.

Lab 3. Group functions, Joins and Subqueries, Set Operators

Goals	• Querying multiple tables using joins • Querying tables using subqueries • Querying tables using set operators
Time	3 hr 30 min

3.1: Group Functions:

1. Display the Highest, Lowest, Total & Average salary of all staff. Label the columns Maximum, Minimum, Total and Average respectively for each Department code. Also round the result to the nearest whole number.

2. Display Department code and number of managers working in that department. Label the column as 'Total Number of Managers' for each department.

3. Get the Department number, and sum of Salary of all non managers where the sum is greater than 20000.

3.2: Joins and Subqueries

1. Write a query which displays Staff Name, Department Code, Department Name, and Salary for all staff who earns more than 20000.

2. Display Staff Code, Staff Name, Department Name, and his manager's number and name. Label the columns Staff#, Staff, Mgr#, Manager.

3. Create a query that will display Student Code, Student Name, Department Name, Subject1, Subject2, and Subject3 for all students who are getting 60 and above in each subject from department 10 and 20.

4. Create a query that will display Student Code, Student Name, Book Code, and Book Name for all students whose expected book return date is today.

5. Create a query that will display Staff Code, Staff Name, Department Name, Designation name, Book Code, Book Name, and Issue Date. For only those staff who have taken any book in last 30 days. . If required, make changes to the table to create such a scenario.

6. Display the unique list of Book code and Book name from the Book transaction.

7. Generate a report which contains the following information.

Staff Code Staff Name Designation Name Department Name
Department Head
For all staff excluding HOD (List should not contain the details of Department head).

8. Generate a report which contains the following information

Student Code Student Name Department Name Total Marks
HOD Name
Sort the output on Department Name and Total Marks.

9. Generate a report which contains the following information.

Staff Code, Staff Name, Designation Name, Department, Book Code, Book Name, Author, Fine For the staff who has not returned the book. Fine will be calculated as Rs. 5 per day.
Fine = 5 * (No. of days = Current Date – Expected return date). Include records in the table to suit this problem statement

10. List Staff Code, Staff Name, and Salary for those who are getting less than the average salary of organization.

11. List the Staff Code, Staff Name who are not Manager.

12. Display Author Name, Book Name for those authors who wrote more than one book.

13. Display Staff Code, Staff Name, and Department Name for those who have taken more than one book.

14. Display top ten students for a specified department. Details are:
Student Code, Student Name, Department Name, Subject1, Subject2, Subject3, Total.

15. a) Display the Staff Name, Department Name, and Salary for those staff who are getting less than average salary in their own department

b) List the details of the staff, experience (in years) whose designations are either PROFESSOR or LECTURER.

16. Create a query that will display the Staff Name, Department Name, and all the staff who work in the same department as a given staff. Give the column as appropriate label.

17. List the Student Code, Student Name for that student who got highest marks in all three subjects in Computer Science department for a particular yearaccording to the table data

18. Display the Student Code, Student Name, and Department Name for that department in which there are maximum number of student are studying.

19. Display Staff Code, Staff Name, Department Name, and Designation name for those who have joined in last 3 months.

20. Display the Manager Name and the total strength of his/her team.

21. Display the details of books that have not been returned and expected return date was last Monday. Book name should be displayed in proper case.. Hint: You can change /add records so that the expected return date suits this problem statement

22. Write a query to display number of people in each Department. Output should display Department Code, Department Name and Number of People.

23. Display Manager Code, Manager Name and salary of lowest paid staff in that manager's team. Exclude any group where minimum salary is less than 10000. Order the result on descending order of salary.

3.3: Set Operators

Observe the following queries and do the given assignments.

```
create table previous_products (
        pid int,
        name varchar2(40),
        unit varchar2(40),
        price int,
        stock int );
```

Example 1: Sample Code

```
create table current_products (
        pid int,
        name varchar2(40),
        unit varchar2(40),
        price int,
        stock int );
```

Example 2: Sample Code

```
insert into previous_products values(7,'Uncle Bob''s Organic Dried Pears','12 - 1 lb
pkgs.',30.00,15);
insert into previous_products values(8,'Northwoods Cranberry Sauce','12 - 12 oz
jars',40.00,6);
insert into previous_products values(1,'Chang','24 - 12 oz bottles',19.00,17);
insert into previous_products values(3,'Aniseed Syrup','12 - 550 ml
bottles',10.00,13);
insert into previous_products values(4,'Chef Anton''s Cajun Seasoning','48 - 6 oz
jars',22.00,53);
insert into previous_products values(5,'Chef Anton''s Gumbo Mix','36
boxes',21.35,0);
insert into previous_products values(6,'Grandma''s Boysenberry Spread','12 - 8 oz
jars',25.00,120);
```

Example 3: Sample Code

```
insert into current_products values(7,'Uncle Bob''s Organic Dried Pears','12 - 1 lb
pkgs.',30.00,15);
insert into current_products values(8,'Northwoods Cranberry Sauce','12 - 12 oz
jars',40.00,6);
insert into current_products values(9,'Mishi Kobe Niku','18 - 500 g pkgs.',97.00,29);
insert into current_products values(10,'Ikura','12 - 200 ml jars',31.00,31);
insert into current_products values(11,'Queso Cabrales','1 kg pkg.',21.00,22);
insert into current_products values(5,'Chef Anton''s Gumbo Mix','36
boxes',21.35,0);
insert into current_products values(6,'Grandma''s Boysenberry Spread','12 - 8 oz
jars',25.00,120);
```

Example 4: Sample Code

1. Get the details of all products irrespective of the fact whether they are in previous set or current set.

2. Get the details of all products along with the repetition of those that were present both in the previous and current sets.

3. Get the details of only those products which were present in the previous set and are still continuing.

4. Get the details of all obsolete products (no longer continued).

Lab 1.Constraints, Adv. Group by, Adv. Subqueries, Managing other db objects.

Goals	Following set of questions are designed to implement the following concepts • Constraints • Adv. Group by • Adv Subqueries
Time	4 hr 30 min

1. Insert the row given below in the Customer table and see the message generated by the Oracle server.

   1002, John, #114 Chicago, #114 Chicago, M, 45, 439525

2. Disable the constraint on CustomerId, and insert the following data:

- 1002, Becker, #114 New York, #114 New york , M, 45, 431525
- 1003, Nanapatekar, #115 India, #115 India , M, 45, 431525

3. Enable the constraint on CustomerId of the Customer table, and see the message generated by the Oracle server.

4. Drop the constraint CustId_Prim on CustomerId and insert the following Data. Alter Customer table, drop constraint Custid_Prim.

- 1002, Becker, #114 New York, #114 New york , M, 45, 431525, 15000.50
- 1003, Nanapatekar, #115 India, #115 India , M, 45, 431525, 20000.50

5. Delete all the existing rows from Customer table, and let the structure remain itself using TRUNCATE statement.
6. Relate AccountsMaster table and CustomerMaster table through CustomerId column with the constraint name Cust_acc.

7. Relate AccountsMaster table and CustomerMaster table through CustomerId column with the constraint name Cust_acc.

Verify that the following tables are present:
COUNTRIES
DEPARTMENTS
EMPLOYEES
JOB_GRADES
JOB_HISTORY
JOBS
LOCATIONS
REGIONS

Determine the validity of the following three statements. Circle either True or False.
1) Group functions work across many rows to produce one result per group.
True/False
2) Group functions include nulls in calculations.
True/False
3) The clause restricts rows before inclusion WHERE in a group calculation.
True/False

The HR department needs the following reports:
4) Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number. Save your SQL statement as lab_05_04.sql. Run the query.

5) Modify the query in lab_05_04.sql to display the minimum, maximum, sum, and average salary for each job type. Save lab_05_04.sql as lab_05_05.sql again.
Run the statement in lab_05_05.sql.

Practice 5-1: Reporting Aggregated Data Using the Group Functions (continued)

6) Write a query to display the number of people with the same job.
Generalize the query so that the user in the HR department is prompted for a job title.
Save the script to a file named lab_05_06.sql. Run the query. Enter IT_PROG when prompted.

7) Determine the number of managers without listing them. Label the column Number of Managers.
Hint: Use the MANAGER_ID column to determine the number of managers.

8) Find the difference between the highest and lowest salaries. Label the column DIFFERENCE.

9) Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

10) Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.

11) Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

Practice 7-1: Using Sub queries to Solve Queries
1) The HR department needs a query that prompts the user for an employee last name.

The query then displays the last name and hires date of any employee in the same department as the employee whose name they supply (excluding that employee). For example, if the user enters, find all employees Zlotkey who work with Zlotkey (excluding Zlotkey).

2) Create a report that displays the employee number, last name, and salary of all employees who earn more than the average salary. Sort the results in order of ascending salary.

3) Write a query that displays the employee number and last name of all employees who work in a department with any employee whose last name contains the letter "u."
Save your SQL statement as lab_07_03.sql. Run your query.

4) The HR department needs a report that displays the last name, department number, and job ID of all employees whose department location ID is 1700.
Modify the query so that the user is prompted for a location ID. Save this to a file named lab_07_04.sql.

5) Create a report for HR that displays the last name and salary of every employee who reports to King.

6) Create a report for HR that displays the department number, last name, and job ID for every employee in the Executive department.

7) Create a report that displays a list of all employees whose salary is more than the salary of any employee from department 60.

8) Modify the query in lab_07_03.sql to display the employee number, last name, and salary of all employees who earn more than the average salary, and who work in a department with any employee whose last name contains a "u." Save lab_07_03.sql as lab_07_08.sql again. Run the statement in lab_07_08.sql.

Lab 2. Privileges, Multitable Inserts, External Tables

Goals	Following set of questions are designed to implement the following concepts • Privileges • Managing objects
Time	4 hr 30 min

1. What privilege should a user be given to log on to the Oracle server? Is this a system privilege or an object privilege?

2. What privilege should a user be given to create tables?

3. If you create a table, who can pass along privileges to other users in your table?

4. You are the DBA. You create many users who require the same system privileges. What should you use to make your job easier?

5. What command do you use to change your password?

6. User21 is the owner of the EMP table and grants the DELETE privilege to User22 by using the WITH GRANT OPTION clause. User22 then grants the DELETE privilege on EMP to User23. User21 now finds that User23 has the privilege and revokes it from User22. Which user can now delete from the EMP
table?

7. You want to grant SCOTT the privilege to update data in the DEPARTMENTS table. You also want to enable SCOTT to grant this privilege to other users. What command do you use?

To complete question 8 and the subsequent ones, you need to connect to the database by using SQL Developer. If you are already not connected, do the following to connect:
1. Click the SQL Developer desktop icon.
2. In the Connections Navigator, use the oraxx account and the corresponding password provided by your instructor to log on to the database.
8. Grant another user query privilege on your table. Then, verify whether that user can use the privilege.
Note: For this exercise, team up with another group. For example, if you are user ora21, team up with another user ora22 .
a. Grant another user privilege to view records in your REGIONS table. Include an option for this user to further grant this privilege to other users.
b. Have the user query your REGIONS table.
c. Have the user pass on the query privilege to a third user (for example, ora23).

Practice 1-1: Controlling User Access (continued)

d. Take back the privilege from the user who performs step b.
Note: Each team can run exercises 9 and 10 independently.
9. Grant another user query and data manipulation privileges on your COUNTRIES table. Make sure that the user cannot pass on these privileges to other users.
10. Take back the privileges on the COUNTRIES table granted to another user.
Note: For exercises 11 through 17, team up with another group.

11. Grant another user access to your DEPARTMENTS table. Have the user grant you query access to his or her DEPARTMENTS table.
12. Query all the rows in your DEPARTMENTS table.
. . .
13. Add a new row to your DEPARTMENTS table. Team 1 should add Education as department number 500 . Team 2 should add Human Resources as department number 510. Query the other team's table.
14. Create a synonym for the other team's DEPARTMENTS table.
15. Query all the rows in the other team's DEPARTMENTS table by using your synonym.

Practice 1-1: Controlling User Access (continued)

16. Revoke the SELECT privilege from the other team.
17. Remove the row that you inserted into the DEPARTMENTS table in step 13 and save the changes.

Practice Solutions 1-1: Controlling User Access
To complete question 8 and the subsequent ones, you need to connect to the database by Using SQL Developer.
1. What privilege should a user be given to log on to the Oracle server? Is this a system or an object privilege?
The CREATES ESSION system privilege
2. What privilege should a user be given to create tables?
The CREATE TABLE privilege
3. If you create a table, who can pass along privileges to other users in your table?
You can, or anyone you have given those privileges to, by using WITH GRANT OPTION
4. You are the DBA. You create many users who require the same system privileges.
What should you use to make your job easier?
Create a role containing the system privileges and grant the role to the users.
5. What command do you use to change your password?
The ALTER USER statement
6. User21 is the owner of the EMP table and grants DELETE privileges to User22 by using the WITH GRANT OPTION clause. User22 then grants DELETE privileges on EMP to User23. User21 now finds that User23 has the privilege and revokes it from User22. Which user can now delete data from the EMP table?
Only User21
7. You want to grant SCOTT the privilege to update data in the DEPARTMENTS table. You also want to enable SCOTT to grant this privilege to other users. What command do you use?
GRANT UPDATE ON departments TO scott WITH GRANT OPTION;

Practice Solutions 1-1: Controlling User Access (continued)
8. Grant another user query privilege on your table. Then, verify whether that user can use the privilege.
Note: For this exercise, team up with another group. For example, if you are user ora21 , team up with another user ora22 .
a) Grant another user privilege to view records in your REGIONS table.
Include an option for this user to further grant this privilege to other users.
Team 1 executes this statement:
```
GRANT select
ON regions
TO <team2_oraxx> WITH GRANT OPTION;
```
b) Have the user query your REGIONS table.
Team 2 executes this statement:
```
SELECT * FROM <team1_oraxx>.regions;
```
c) Have the user pass on the query privilege to a third user (for example, ora23).

Team 2 executes this statement.
GRANT select
ON <team1_oraxx>.regions
TO <team3_oraxx>;
d) Take back the privilege from the user who performs step b.
Team 1 executes this statement.
REVOKE select
ON regions
FROM <team2_oraxx>;
9. Grant another user query and data manipulation privileges on your COUNTRIES table.
Make sure the user cannot pass on these privileges to other users.
Team 1 executes this statement.
GRANT select, update, insert
ON COUNTRIES
TO <team2_oraxx>;

Practice 2-1: Managing Schema Objects
In this practice, you use the ALTER TABLE command to modify columns and add constraints.
You use the CREATE INDEX command to create indexes when creating a table, along with
the CREATE TABLE
Command . You create external tables.
1. Create the DEPT2 table based on the following table instance chart. Enter the syntax in the
SQL Worksheet. Then, execute the statement to create the table. Confirm that the table is
created.
Column Name ID NAME
Key Type
Nulls/Unique
FK Table
FK Column
Data type NUMBER VARCHAR2
Length 7 25
2. Populate the DEPT2 table with data from the DEPARTMENTS table. Include only the
columns that you need.
3. Create the EMP2 table based on the following table instance chart. Enter the syntax in
the SQL Worksheet. Then execute the statement to create the table. Confirm that the table is
created.
Column Name ID LAST_NAME FIRST_NAME DEPT_ID
Key Type
Nulls/Unique
FK Table
FK Column
Data type NUMBER VARCHAR2 VARCHAR2 NUMBER
Length 7 25 25 7

Practice 2-1: Managing Schema Objects (continued)
4. Modify the EMP2t able to allow for longer employee last names. Confirm your modification.
5. Create the EMPLOYEES2t able based on the structure of the EMPLOYEES table.
Include only the EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY, and
DEPARTMENT_ID columns. Name the columns in your new table ID, FIRST_NAME,
LAST_NAME, SALARY, and DEPT_ID, respectively.
6. Drop the EMP2 table.
7. Query the recycle bin to see whether the table is present.
8. Restore the EMP2t able to a state before the DROP statement.

9. Drop the FIRST_NAME column from the EMPLOYEES2t able. Confirm your modification by checking the description of the table.

Practice 2-1: Managing Schema Objects (continued)
10. In the EMPLOYEES2t able, mark the DEPT_ID column as UNUSED. Confirm your modification by checking the description of the table.
11. Drop all the UNUSED columns from the EMPLOYEES2t able. Confirm your modification by checking the description of the table.
12. Add a table-level PRIMARY KEY constraint to the EMP2t able on the ID column. The constraint should be named at creation. Name the constraint my_emp_id_pk .
13. Create a PRIMARY KEY constraint to the DEPT2 table using the ID column. The constraint should be named at creation. Name the constraint my_dept_id_pk .
14. Add a foreign key reference on the EMP2 table that ensures that the employee is not assigned to a nonexistent department. Name the constraint my_emp_dept_id_fk .
15. Modify the EMP2t able. Add a COMMISSION column of the NUMBER data type, precision 2, scale 2. Add a constraint to the COMMISSION column that ensures that a commission value is greater than zero.
16. Drop the EMP2 and DEPT2 tables so that they cannot be restored. Verify the recycle bin.
17. Create the DEPT_NAMED_INDEX table based on the following table instance chart. Name the index for the PRIMARY KEY column as DEPT_PK_IDX.
Column Name Deptno Dname
Primary Key Yes
Data Type Number VARCHAR2
Length 4 30
18. Create an external table library_items_ext . Use the ORACLE_LOADER access driver.

Practice 2-1: Managing Schema Objects (continued)
Note: The emp_dir directory and library_items.dat file are already created for this exercise.
Library_items.dat has records in the following format:
2354, 2264, 13.21, 150,
2355, 2289, 46.23, 200,
2355, 2264, 50.00, 100,
a. Open the lab_02_18.sql file. Observe the code snippet to create the library_items_ext external table. Then replace < TODO1 > , < TODO2 > ,
< TODO3 > , and < TODO4 > as appropriate and save the file as
lab_02_18_soln.sql . Run the script to create the external table.
b. Query the library_items_ext table.
19. The HR department needs a report of the addresses of all departments. Create an external table as dept_add_ext using the ORACLE_DATAPUMP access driver.
The report should show the location ID, street address, city, state or province, and country in the output. Use a NATURALJ OIN to produce the results.
Note: The emp_dir directory is already created for this exercise.
a. Open the lab_02_19.sql file. Observe the code snippet to create the dept_add_ext external table. Then, replace < TODO1 >, < TODO2 >, and < TODO3 > with the appropriate code.
Replace < oraxx_emp4.exp > and < oraxx_emp5.exp > with the appropriate file names. For example, if you are
the ora21 user, your file names are ora21_emp4.exp and ora21_emp5.exp . Save the script as lab_02_19_soln.sql .
b. Run the lab_02_19_soln.sql script to create the external table.
c. Query the dept_add_ext table.

Practice 2-1: Managing Schema Objects (continued)
Note: When you perform the preceding step, two files oraxx_emp4.exp and oraxx_emp5.exp are created under the default directory emp_dir .

20. Create the emp_books table and populate it with data. Set the primary key as deferred and observe what happens at the end of the transaction.
a. Run the lab_02_20_a.sql file to create the emp_books table. Observe that the emp_books_pk primary key is not created as deferrable.
b. Run the lab_02_20_b.sql file to populate data into the emp_books table.
What do you observe?
c. Set the emp_books_pk constraint as deferred. What do you observe?
d. Drop the emp_books_pk constraint.
e. Modify the emp_books table definition to add the emp_books_pk constraint as deferrable this time.

Practice 2-1: Managing Schema Objects (continued)
f. Set the emp_books_pk constraint as deferred.
g. Run the lab_02_20_g.sql file to populate data into the emp_books table.
What do you observe?
h. Commit the transaction. What do you observe?

Practice 3-1: Managing Objects with Data Dictionary Views
In this practice, you query the dictionary views to find information about objects in your schema.
1. Query the USER_TABLES data dictionary view to see information about the tables that you own.
…
2. Query the ALL_TABLES data dictionary view to see information about all the tables that you can access. Exclude the tables that you own.
Note: Your list may not exactly match the following list:
…
3. for a specified table, create a script that reports the column names, data types, and data types ' lengths, as well as whether nulls are allowed. Prompt the user to enter the table name. Give appropriate aliases to the DATA_PRECISION and DATA_SCALE columns. Save this script in a file named lab_03_01.sql.
For example, if the user enters DEPARTMENTS, the following output results:
4. Create a script that reports the column name, constraint name, constraint type, search Condition and status for a specified table. You must join the USER_CONSTRAINTS and USER_CONS_COLUMNS tables to obtain all this information. Prompt the user to enter the table name. Save the script in a file named lab_03_04.sql.
For example, if the user enters DEPARTMENTS, the following output results:
5. Add a comment to the DEPARTMENTS table. Then query the USER_TAB_COMMENTS view to verify that the comment is present.
6. Create a synonym for your EMPLOYEES table. Call it EMP. Then find the names of all synonyms that are in your schema.
7. Run lab_03_07.sql to create the dept50 view for this exercise.
You need to determine the names and definitions of all the views in your schema.
Create a report that retrieves view information: the view name and text from the USER_VIEWS data dictionary view.
Note: The EMP_DETAILS_VIEW was created as part of your schema.
Note: You can see the complete definition of the view if you use Run Script (or press F5) in SQL Developer. If you use Execute Statement (or press F9) in SQL Developer, scroll horizontally in the result pane. If you use SQL*Plus, to see more contents of a LONG column, use the SET LONG n command, where n is the value of the number of characters of the LONG column that you want to see.
8. Find the names of your sequences. Write a query in a script to display the following information about your sequences: sequence name, maximum value, increment size, and last number. Name the script lab_03_08.sql. Run the statement in your script.

Run the lab_03_09_tab.sql script as a prerequisite for exercises 9 through 11.
Alternatively, open the script file to copy the code and paste it into your SQL Worksheet.
Then execute the script. This script:
• Drops if there are existing tables DEPT2 and EMP2
• Creates the DEPT2 and EMP2 tables
Note: In Practice 2, you should have already dropped the DEPT2 and EMP2 tables so that they cannot be restored.
9. Confirm that both the DEPT2 and EMP2 tables are stored in the data dictionary.
10. Confirm that the constraints were added by querying the USER_CONSTRAINTS view.
Note the types and names of the constraints.
11. Display the object names and types from the USER_OBJECTS data dictionary view for the EMP2
and DEPT2 tables.
12. Create the SALES_DEPT table based on the following table instance chart. Name the index for the PRIMARY KEY column SALES_PK_IDX . Then query the data dictionary view to find the index name, table name, and whether the index is unique.

Practice 3-1: Managing Objects with Data Dictionary Views(continued)

Column Name	Team_Id	Location
Primary Key	Yes	
Data Type	Number	VARCHAR2
Length	3	30

Lab 3. Merge statements , Time Zones

Goals	Following set of questions are designed to implement the following concepts • Merge statements • Time Zones
Time	4 hr 30 min

Practice 4-1: Manipulating Large Data Sets
In this practice, you perform multitable INSERT and MERGE operations, and track row versions.
1. Run the lab_04_01.sql script in the lab folder to create the SAL_HISTORY table.
2. Display the structure of the SAL_HISTORY table.
3. Run the lab_04_03.sql script in the lab folder to create the MGR_HISTORY table.
4. Display the structure of the MGR_HISTORY table.
5. Run the lab_04_05.sql script in the lab folder to create the SPECIAL_SAL table.
6. Display the structure of the SPECIAL_SAL table.
7. a. Write a query to do the following:
- Retrieve details such as the employee ID, hire date, salary, and manager ID of those employees whose employee ID is less than 125 from the EMPLOYEES table.
- If the salary is more than $20,000, insert details such as the employee ID and salary into the SPECIAL_SAL table.

Practice 4-1: Manipulating Large Data Sets (continued)
- Insert details such as the employee ID, hire date, and salary into the SAL_HISTORY table.
- Insert details such as the employee ID, manager ID, and salary into the MGR_HISTORY table.
b. Display the records from the SPECIAL_SAL table.
c. Display the records from the SAL_HISTORY table.
d. Display the records from the MGR_HISTORYta ble.

8.
a. Run the lab_04_08a.sql script in the lab folder to create the SALES_WEEK_DATA table.
b. Run the lab_04_08b.sql script in the lab folder to insert records into the SALES_WEEK_DATAta ble.

Practice 4-1: Manipulating Large Data Sets (continued)
c. Display the structure of the SALES_WEEK_DATA table.
d. Display the records from the SALES_WEEK_DATA table.
e. Run the lab_04_08_e.sql script in the lab folder to create the EMP_SALES_INFO table.
f. Display the structure of the EMP_SALES_INFO table.
g. Write a query to do the following:
- Retrieve details such as ID, week ID, sales quantity on Monday, sales quantity on Tuesday, sales quantity on Wednesday, sales quantity on Thursday, and sales quantity on Friday from the SALES_WEEK_DATA table.
- Build a transformation such that each record retrieved from the SALES_WEEK_DATAt able is converted into multiple records for the EMP_SALES_INFO table.
Hint: Use a pivoting INSERT statement.
h. Display the records from the EMP_SALES_INFO table.

9. You have the data of past employees stored in a flat file called emp.data. You want to store the names and email IDs of all employees, past and present, in a table. To do this, first create an external table called EMP_DATA using the emp.dat source file in the emp_dir directory. Use the lab_04_09.sql script to do this.
10. Next, run the lab_04_10.sql script to create the EMP_HIST table.
a. Increase the size of the email column to 45.
b. Merge the data in the EMP_DATA table created in the last lab into the data in the EMP_HIST table. Assume that the data in the external EMP_DATA table is the most up-to-date. If a row in the EMP_DATA table matches the EMP_HIST table, update the email column of the EMP_HIST table to match the EMP_DATA table row. If a row in the EMP_DATA table does not match, insert it into the EMP_HIST table. Rows are considered matching when the employee's first and last names are identical.
c. Retrieve the rows from EMP_HIST after the merge.
11. Create the EMP3 table by using the lab_04_11.sql script. In the EMP3 table, change the department for Kochhar to 60 and commit your change. Next, change the department for Kochhar to 50 and commit your change. Track the changes to Kochhar by using the Row Versions feature.

Practice 5-1: Managing Data in Different Time Zones
In this practice, you display time zone offsets, CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP. You also set time zones and use the EXTRACT function.
1. Alter the session to set NLS_DATE_FORMATto DD-MON-YYYYH H24: MI:SS .
2. a. Write queries to display the time zone offsets (TZ_OFFSET) for the following time zones.
- US/Pacific-New
- Singapore
- Egypt
b. Alter the session to set the TIME_ZONE parameter value to the time zone offset of US/Pacific-New.
c. Display CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP for this session.
d. Alter the session to set the TIME_ZONE parameter value to the time zone offset of Singapore.
e. Display CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP for this session.
Note: The output might be different based on the date when the command is executed.
Note: Observe in the preceding practice that CURRENT_DATE, CURRENT_TIMESTAMP, and LOCALTIMESTAMP are sensitive to the session time zone.
3. Write a query to display DBTIMEZONE and SESSIONTIMEZONE .
4. Write a query to extract the YEAR from the HIRE_DATE column of the EMPLOYEES table for those employees who work in department 80.

Practice 5-1: Managing Data in Different Time Zones (continued)
5. Alter the session to set NLS_DATE_FORMAT to DD-MON-YYYY.
6. Examine and run the lab_05_06.sql script to create the SAMPLE_DATES table and populate it.
a. Select from the table and view the data.
b. Modify the data type of the DATE_COL column and change it to TIMESTAMP. Select from the table to view the data.
c. Try to modify the data type of the DATE_COL column and change it to TIMESTAMP WITH TIME ZONE . What happens?
7. Create a query to retrieve last names from the EMPLOYEES table and calculate the review status. If the year hired was 1998, display needs Review for the review status; otherwise, display not this year! Name the review status column Review. Sort the results by the HIRE_DATE column.
Hint: Use a CASE expression with the EXTRACT function to calculate the review status.

…
Practice 5-1: Managing Data in Different Time Zones (continued)
8. Create a query to print the last names and the number of years of service for each employee. If the employee has been employed for five or more years, print 5 years of service . If the employee has been employed for 10 or more years, print 10 years of service . If the employee has been employed for 15 or more years, print 15 years of service . If none of these conditions match, print may be
next year! Sort the results by the HIRE_DATE column. Use the EMPLOYEES table.
Hint: Use CASE expressions and TO_YMINTERVAL.
...

Practice 6-1: Retrieving Data by Using Subqueries
In this practice, you write multiple-column subqueries, and correlated and scalar subqueries. You also solve problems by writing the WITH clause.
1. Write a query to display the last name, department number, and salary of any employee whose department number and salary both match the department number and salary of any employee who earns a commission.
2. Display the last name, department name, and salary of any employee whose salary and commission match the salary and commission of any employee located in location ID 1700.
3. Create a query to display the last name, hire date, and salary for all employees who have the same salary and commission as Kochhar.
Note: Do not display Kochhar in the result set.
4. Create a query to display the employees who earn a salary that is higher than the salary of all the sales managers (JOB_ID = 'SA_MAN'). Sort the results from the highest to the lowest.

Practice 6-1: Retrieving Data by Using Subqueries (continued)
5. Display details such as the employee ID, last name, and department ID of those employees who live in cities the names of which begin with T.
6. Write a query to find all employees who earn more than the average salary in their departments. Display last name, salary, department ID, and the average salary for the department. Sort by average salary and round to two decimals. Use aliases for the columns retrieved by the query as shown in the sample output.
7. Find all employees who are not supervisors.
a. First, do this using the NOT EXISTS operator.

Practice 6-1: Retrieving Data by Using Subqueries (continued)
b. Can this be done by using the NOT IN operator? How, or why not?
8. Write a query to display the last names of the employees who earn less than the average salary in their departments.
9. Write a query to display the last names of the employees who have one or more coworkers in their departments with later hire dates but higher salaries.
10. Write a query to display the employee ID, last names, and department names of all the employees.
Note: Use a scalar subquery to retrieve the department name in the SELECT statement.
…
Oracle Database 11 g: SQL Fundamentals II A - 64
Practice 6-1: Retrieving Data by Using Subqueries (continued)
11. Write a query to display the department names of those departments whose total salary cost is above one-eighth (1/8) of the total salary cost of the whole company.
Use the WITH clause to write this query. Name the query SUMMAR.Y

Lab 4. Regular Expression

Goals	Following set of questions are designed to implement the following concepts • Regular Expression
Time	2 hr 30 min

Practice 7-1: Regular Expression Support

```
SQL> CREATE VIEW v AS SELECT 'www.oracle-developer.net' AS string FROM   dual;

View created.

SQL> SELECT * FROM v;

STRING
------------------------
www.oracle-developer.net

1 row selected.
```

Write a query to find the number of occurrences of "e" in this website's address given in view V.

1.2.2: See the given snippet:

```
SQL> SELECT REGEXP_SUBSTR(string, '\.[a-z-]+\.') AS url_middle_10g FROM   v;

URL_MIDDLE_10G
------------------
.oracle-developer.

1 row selected.
```

Generate the output as shown bewlo without using the further functions such as REPLACE, LTRIM, RTRIM or even SUBSTR in the given query.

```
URL_MIDDLE_CLEANED_10G
----------------------
oracle-developer

1 row selected.
```

In this practice, you use regular expressions functions to search for, replace, and manipulate data. You also create a new CONTACTS table and add a CHECK constraint to the p_number column to ensure that phone numbers are entered into the database in a specific standard format.

1. Write a query to search the EMPLOYEES table for all the employees whose first names start with “Ki” or “Ko.”

2. Create a query that removes the spaces in the STREET_ADDRESS column of the LOCATIONS table in the display. Use “ Street Address ” as the column heading.
3. Create a query that displays “ S t” replaced by “ Street ” in the STREET_ADDRESS column of the LOCATIONS table. Be careful that you do not affect any rows that already have “Street ” in them. Display only those rows that are affected.
4. Create a contacts table and add a check constraint to the p_number column to enforce the following format mask to ensure that phone numbers are entered into the database in the following standard format: (XXX) XXX-XXXX . The table should
have the following columns:
- l_name varchar2(30)
- p_number varchar2 (30)

Oracle Database 11 g : SQL Fundamentals II A - 70
Practice 7-1: Regular Expression Support (continued)

5. Run the SQL script lab_07_05.sql to insert the following seven phone numbers
into the contacts table. Which numbers are added?

l_name Column Value	p_number Column Value
NULL	‘(650) 555-5555’
NULL	‘(215) 555-3427’
NULL	‘650 555-5555’
NULL	‘650 555 5555’
NULL	‘650-555-5555’
NULL	‘(650)555-5555’
NULL	‘ (650) 555-5555’

6. Write a query to find the number of occurrences of the DNA pattern ctc in the string gtctcgtctcgttctgtctgtcgttctg . Ignore case-sensitivity.

Appendices

Appendix A: DBMS SQL Standards

Key points to keep in mind:

NA

How to follow DBMS SQL standards:

- Decide upon a database naming convention, standardize it across your organization and be consistent in following it. It helps make your code more readable and understandable.
- Tables:
  - Tables represent the instances of an entity. For example, you store all your customer information in a table. Here “customer” is an entity and all the rows in the customers table represent the instances of the entity “customer”. So name your table using the entity it represents, namely “Customer”. Since the table is storing “multiple instances” of customers, make your table name a plural word.
  - Keeping this in mind:
    - name your customer table as “Customers'”
    - name your order storage table as “Orders”
    - name your error messages table as “ErrorMessages”
  - Suppose your database deals with different logical functions and you want to group your tables according to the logical group they belong to. It will help prefixing your table name with a two or three character prefix that can identify the group.

    For example: Your database has tables which store information about Sales and Human resource departments, then you can name all your tables related to Sales department as shown below:
    - SL_NewLeads
    - SL_Territories
    - SL_TerritoriesManagers

    You can name all your tables related to Human resources department as shown below:
    - HR_Candidates
    - HR_PremierInstitutes
    - HR_InterviewSchedules
  - Prefix the table names with owner names, as this improves readability, and avoids any unnecessary confusion.
- Primary keys:

    - Primary key is the column(s) that can uniquely identify each row in a table. So just use the column name prefixed with “pk_” + “Table name” for naming primary keys.
    - Given below is an example of how we can name the primary key on the CustomerID column of Customers table:
        - pk_Customers_CustomerID
    - Consider concatenating the column names in case of composite primary keys.
- Foreign keys:
    - Foreign keys are used to represent the relationships between tables which are related. So a foreign key can be considered as a link between the “column of a referencing table” and the “primary key column of the referenced table”.
    - We can use the following naming convention for foreign keys:
        - fk_referencing table + referencing column_referenced table + referenced column
    - Based on the above convention, we can name the foreign key, which references the CustomerID column of the Customers table from the Order’s tables CustomerID column as shown below:
        - fk_OrdersCustomerID_CustomersCustomerID
    - Foreign key can be composite too. In that case, consider concatenating the column names of referencing and referenced tables while naming the foreign key. This might make the name of the foreign key lengthy. However, you should not be worried about it, as you will never reference this name from your code, except while creating/dropping these constraints.
- Default and Check constraints:
    - Use the column name to which the defaults /check constraints are bound to. Prefix it with “def” and “chk” prefixes respectively for Default and Check constraints.
    - We can name the default constraint for OrderDate Column as def_OrderDate, and the check constraint for OrderDate column as chk_OrderDate.
- Do not use any reserved words for naming my database objects, as that can lead to some unpredictable situations.
- Do not depend on undocumented functionality. The reasons are:
    - you will not get support, when something goes wrong with your undocumented code
    - undocumented functionality is not guaranteed to exist (or behave the same) in a future release or service pack, thereby breaking your code
- Make sure you normalize your data at least till third normal form. At the same time, do not compromise on query performance. A little de-normalization helps queries perform faster.

Appendix B: Coding Best Practices

Given below are a few best practices in DBMS SQL:

- Do not use SELECT * in your queries. Always write the required column names after the SELECT statement, as shown below:

  ```
  SELECT CustomerID, CustomerFirstName, City
  ```

  This technique results in less disk IO, less network traffic, and hence better performance.

- Try to avoid wildcard characters at the beginning of a word while searching by using the LIKE keyword. This is because this arrangement results in an index scan, which is defeating the purpose of having an index. The following statement results in an index scan, while the second statement results in an index seek:

  ```
  1. SELECT LocationID FROM Locations WHERE Specialities LIKE
  '%pples'
  2. SELECT LocationID FROM Locations WHERE Specialities LIKE
  'A%s'
  ```

  Also avoid searching with not equals operators (<> and NOT) as they result in table and index scans. If you must do heavy text-based searches, consider using the Full-Text search feature of SQL Server for better performance.

- Use “Derived tables” wherever possible, as they perform better. Consider the following query to find the second highest salary from Staff table:

  ```
  SELECT MIN(Salary)
  FROM Staff
  WHERE EmpID IN
  (
  SELECT TOP 2 EmpID
  FROM Staff
  ORDER BY Salary Desc
  )
  ```

  The same query can be re-written by using a derived table as shown below, and it performs twice as fast as the above query:

  ```
  SELECT MIN(Salary)
  FROM
  (
  SELECT TOP 2 Salary
  FROM Staff
  ORDER BY Salary Desc
  ) AS A
  ```

  This is just an example. The results might differ in different scenarios depending upon the database design, indexes, volume of data, etc. So test all the possible ways a query can be written and go with the efficient one.

- Use char data type for a column, only when the column is non-nullable. If a char column is nullable, it is treated as a fixed length column in SQL Server 7.0+. So a char(100), when NULL, will eat up 100 bytes, resulting in space wastage. So use varchar(100) in this situation. Of course, variable length columns do have a very little processing overhead over fixed length columns. Carefully choose between char and varchar depending upon the length of the data you are going to store.

- Always use a column list in your INSERT statements. This helps in avoiding problems when the table structure changes (like adding a column).

- Do not use the column numbers in the ORDER BY clause as it impairs the readability of the SQL statement. Further, changing the order of columns in the SELECT list has no impact on the ORDER BY when the columns are referred by names instead of numbers. Consider the following example, in which the second query is more readable than the first one:

```
SELECT OrderID, OrderDate
FROM Orders
ORDER BY 2

SELECT OrderID, OrderDate
FROM Orders
ORDER BY OrderDate
```

Appendix C: Table of Examples

Appendix D: Debugging Examples

1) Identify the mistake in the query (if any) in terms of computed value.
   - SELECT empno, ename, sal+comm as Total FROM emp;

2) CREATE TABLE ITEM_MASTER
   (ITEM_NO VARCHAR2(4) PRIMARY KEY,ITEM_DESC VARCHAR2(25));

```
INSERT INTO ITEM_MASTER VALUES ('1001', 'pen');
INSERT INTO ITEM_MASTER VALUES ('1002', 'Pencil  ');
INSERT INTO ITEM_MASTER VALUES ('1003', 'Eracer');
INSERT INTO ITEM_MASTER VALUES ('1104', 'Keyboard MF');
INSERT INTO ITEM_MASTER VALUES ('1005', 'Gel');
INSERT INTO ITEM_MASTER VALUES ('1006', 'chart');
INSERT INTO ITEM_MASTER VALUES ('1007', 'Map');
INSERT INTO ITEM_MASTER VALUES ('1008', 'note book');
INSERT INTO ITEM_MASTER VALUES ('1009', 'STAPLER');
INSERT INTO ITEM_MASTER VALUES ('1010', '  story book');
INSERT INTO ITEM_MASTER VALUES ('1011', 'Calculator');
INSERT INTO ITEM_MASTER VALUES ('2012', 'Writing Pad(S)');
INSERT INTO ITEM_MASTER VALUES ('1013', 'Geometry box');
INSERT INTO ITEM_MASTER VALUES ('1014', 'Id card holder');
INSERT INTO ITEM_MASTER VALUES ('1015', 'FILE');
INSERT INTO ITEM_MASTER VALUES ('1016', 'Bag');
INSERT INTO ITEM_MASTER VALUES ('1017', 'Mobile Pouch');
INSERT INTO ITEM_MASTER VALUES ('1018', 'Punching Machine');
```

a) Correct the mistake in the below query to get the list in ascending order of item_no:
SELECT item_no, item_desc FROM item_master order by item_no;

b) Identify the mistake, if any to get the list of records correctly:
 SELECT item_no, item_desc FROM item_master WHERE item_desc like 'k%' or like 'p%' or item_desc like 'S%';

Hint: use Conversion Functions for above Qs

3) Rewrite the subquery below to correct the errors(if any)

```
SELECT staff_name,dept_name,staff_sal
      FROM  staff_master s,department_master d
       WHERE s.dept_code = sm.dept_code AND   --d
          staff_sal < ALL (SELECT AVG(staff_sal) FROM staff_master sm
WHERE department_code = s.dept_code);
```

4) Complete the below query to increase the salary for those people whose salary is less than their Department's average

```
 ---
UPDATE emp a
SET sal = (select avg(sal) FROM enmp b where
....your query....
)
WHERE SAL <
....your query...
```