Revature Associate SQL Workbook

Oracle 11g

Working with Relational Database Management Systems

Contents

[1.0 Setting up Chinnook 4](#_Toc387140451)

[2.0 SQL Queries 4](#_Toc387140452)

[2.1 SELECT 5](#_Toc387140453)

[2.2 ORDER BY 5](#_Toc387140454)

[2.3 INSERT INTO 5](#_Toc387140455)

[2.4 UPDATE 5](#_Toc387140456)

[2.5 LIKE 5](#_Toc387140457)

[2.6 BETWEEN 5](#_Toc387140458)

[2.7 DELETE 5](#_Toc387140459)

[3.0 SQL Functions 5](#_Toc387140460)

[3.1 System Defined Functions 5](#_Toc387140461)

[3.2 System Defined Aggregate Functions 6](#_Toc387140462)

[3.3 User Defined Scalar Functions 6](#_Toc387140463)

[3.4 User Defined Table Valued Functions 6](#_Toc387140464)

[4.0 Stored Procedures 6](#_Toc387140465)

[4.1 Basic Stored Procedure 6](#_Toc387140466)

[4.2 Stored Procedure Input Parameters 6](#_Toc387140467)

[4.3 Stored Procedure Output Parameters 6](#_Toc387140468)

[5.0 Transactions 6](#_Toc387140469)

[6.0 Triggers 6](#_Toc387140470)

[6.1 AFTER/FOR 6](#_Toc387140471)

[6.2 INSTEAD OF 7](#_Toc387140472)

[7.0 JOINS 7](#_Toc387140473)

[7.1 INNER 7](#_Toc387140474)

[7.2 OUTER 7](#_Toc387140475)

[7.3 RIGHT 7](#_Toc387140476)

[7.4 CROSS 7](#_Toc387140477)

[7.5 SELF 7](#_Toc387140478)

[8.0 Indexes 7](#_Toc387140479)

[8.1 Clustered Indexes 7](#_Toc387140480)

[9.0 Administration 7](#_Toc387140481)

[1.0 Creating the OfficeSupply Database 8](#_Toc387140482)

[1.1 Create Company Database using SSMS Interface 8](#_Toc387140483)

[1.2 Create Company Database using DDL 8](#_Toc387140484)

[2.0 Creating Tables and Relationships 8](#_Toc387140485)

[2.1 Create Tables for OfficeSupply 8](#_Toc387140486)

[2.2 Creating Relationships 9](#_Toc387140487)

[3.0 Performing SQL Queries 9](#_Toc387140488)

[3.1 SELECT 10](#_Toc387140489)

[3.2 ORDER BY 11](#_Toc387140490)

[3.3 INSERT INTO 11](#_Toc387140491)

[3.4 UPDATE 11](#_Toc387140492)

[3.5 LIKE 11](#_Toc387140493)

[3.6 BETWEEN 11](#_Toc387140494)

[3.7 DELETE 11](#_Toc387140495)

[4.0 SQL Functions 11](#_Toc387140496)

[4.1 System Defined Scalar Functions 11](#_Toc387140497)

[4.2 System Defined Aggregate Function 11](#_Toc387140498)

[4.3 User Defined Scalar Functions 12](#_Toc387140499)

[4.4 User Defined Table Valued Functions 12](#_Toc387140500)

[5.0 Stored Procedures 12](#_Toc387140501)

[5.1 Basic Stored Procedure 12](#_Toc387140502)

[5.2 Stored Procedure Input Parameters 12](#_Toc387140503)

[5.3 Stored Procedure Output Parameters 12](#_Toc387140504)

[6.0 Transactions 12](#_Toc387140505)

[7.0 Triggers 13](#_Toc387140506)

[7.1 AFTER/FOR Triggers 13](#_Toc387140507)

[7.2 INSTEAD OF Triggers 13](#_Toc387140508)

[8.0 JOINS 13](#_Toc387140509)

[8.1 INNER JOIN 13](#_Toc387140510)

[8.2 OUTER JOIN 13](#_Toc387140511)

[8.3 RIGHT JOIN 13](#_Toc387140512)

[8.4 LEFT JOIN 13](#_Toc387140513)

[8.5 CROSS JOIN 13](#_Toc387140514)

[8.6 SELF-JOIN 13](#_Toc387140515)

[9.0 Views 14](#_Toc387140516)

[10.0 Indexes 14](#_Toc387140517)

[10.1 Clustered Indexes 14](#_Toc387140518)

[11.0 Administration 14](#_Toc387140519)

Part I – Working with an existing database

# Setting up Oracle Chinook

In this section you will begin the process of working with the Oracle Chinook database

Task – Open the Chinook\_Oracle.sql file and execute the scripts within.

## 2.0 SQL Queries 14

In this section you will be performing various queries against the Oracle Chinook database.

## 2.1 SELECT

Task – Select all records from the Employee table.

Task – Select all records from the Employee table where last name is King.

Task – Select all records from the Employee table where first name is Andrew and REPORTSTO is NULL.

## 2.2 ORDER BY

Task – Select all albums in Album table and sort result set in descending order by title.

Task – Select first name from Customer and sort result set in ascending order by city

## 2.3 INSERT INTO

Task – Insert two new records into Genre table

Task – Insert two new records into Employee table

Task – Insert two new records into Customer table

## 2.4 UPDATE

Task – Update Aaron Mitchell in Customer table to Robert Walter

Task – Update name of artist in the Artist table “Creedence Clearwater Revival” to “CCR”

## 2.5 LIKE

Task – Select all invoices with a billing address like “T%”

## 2.6 BETWEEN

Task – Select all invoices that have a total between 15 and 50

Task – Select all employees hired between 1st of June 2003 and 1st of March 2004

## 2.7 DELETE

Task – Delete a record in Customer table where the name is Robert Walter (There may be constraints that rely on this, find out how to resolve them).

# SQL Functions 6

In this section you will be using the Oracle system functions, as well as your own functions, to perform various actions against the database

## 3.1 System Defined Functions

Task – Create a function that returns the current time.

Task – create a function that returns the length of a mediatype from the mediatype table

## 3.2 System Defined Aggregate Functions

Task – Create a function that returns the average total of all invoices

Task – Create a function that returns the most expensive track

## 3.3 User Defined Scalar Functions

Task – Create a function that returns the average price of invoiceline items in the invoiceline table

## 3.4 User Defined Table Valued Functions

Task – Create a function that returns all employees who are born after 1968.

# 4.0 Stored Procedures 4

In this section you will be creating and executing stored procedures. You will be creating various types of stored procedures that take input and output parameters.

## 4.1 Basic Stored Procedure

Task – Create a stored procedure that selects the first and last names of all the employees.

## 4.2 Stored Procedure Input Parameters

Task – Create a stored procedure that updates the personal information of an employee.

Task – Create a stored procedure that returns the managers of an employee.

## 4.3 Stored Procedure Output Parameters

Task – Create a stored procedure that returns the name and company of a customer.

# 5.0 Transactions 2

In this section you will be working with transactions. Transactions are usually nested within a stored procedure.

Task – Create a transaction that given a invoiceId will delete that invoice (There may be constraints that rely on this, find out how to resolve them).

Task – Create a transaction nested within a stored procedure that inserts a new record in the Customer table

# 6.0 Triggers

In this section you will create various kinds of triggers that work when certain DML statements are executed on a table.

## 6.1 AFTER/FOR 3

Task - Create an after insert trigger on the employee table fired after a new record is inserted into the table.

Task – Create an after update trigger on the album table that fires after a row is inserted in the table

Task – Create an after delete trigger on the customer table that fires after a row is deleted from the table.

# 7.0 JOINS 5

In this section you will be working with combing various tables through the use of joins. You will work with outer, inner, right, left, cross, and self joins.

## 7.1 INNER

Task – Create an inner join that joins customers and orders and specifies the name of the customer and the invoiceId.

## 7.2 OUTER

Task – Create an outer join that joins the customer and invoice table, specifying the CustomerId, firstname, lastname, invoiceId, and total.

## 7.3 RIGHT

Task – Create a right join that joins album and artist specifying artist name and title.

## 7.4 CROSS

Task – Create a cross join that joins album and artist and sorts by artist name in ascending order.

## 7.5 SELF

Task – Perform a self-join on the employee table, joining on the reportsto column.

Part II – Creating and working with your own custom database

# Creating the OfficeSupply Database

Objective: In this section you will be creating a database based on information given to you in this handbook.

## 1.1 Create Company Database using SSMS Interface

Task – Create a user and name it “OfficeSupply” in Oracle Web Console

Task – Delete the OfficeSupply user

## 1.2 Create Company Database using DDL

Task – Create a user and name it “OfficeSupply” using DDL (SQL Script in SQL Developer)

# Creating Tables and Relationships

Objective: In this section you will be creating tables for the OfficeSupply database, you will create attributes and corresponding datatypes. You will also create relationships between the tables.

## 2.1 Create Tables for OfficeSupply

Task – Using the DDL, create a table named “Employees” with following attributes and datatypes:

EmployeeID(PK number, not null), UserName(varchar(20), not null), Password(varchar(20), not null),

Name(varchar(25), not null), Department(char(2), not null), Manager(number, not null).

Task – Using the DDL, create a table named “Orders” with following attributes and datatypes:

OrderID(PK, number, not null), EmployeeID(FK, number, not null), OrderDate(date, not null), Status(char, not null).

Task – Using the DDL, create a table named “OrderItem” with the following attributes and datatypes:

OrderID(PK, FK, number, not null), ProductID(PK, FK, number, not null), Quantity(number, not null).

Task – Using DDL, create a table named “Category” with the following attributes and datatypes:

CatID(PK, number, not null), Name(varchar(80), null), Descript(varchar(255), null)

Task – Using DDL, create a table named “Product” with the following attributes and datatypes:

ProductID(PK, number, not null), CatID(FK, number, not null), Name(varchar(80), null), Descript(varchar(255), null),

UnitCost(number, null), SuppID(FK, number, not null).

Task – Using DDL, create a table named “Supplier” with the following attributes and datatypes:

SuppID(PK, number, not null), Name(varchar(80), null).

## 2.2 Creating Relationships

Task – Create a 1:N relationship between Employees(PK) and Orders(FK)

Task – Create a 1:N relationship between Orders(PK) and OrderItem(FK)

Task – Create a 1:N relationship between Product(PK) and OrderItem(FK)

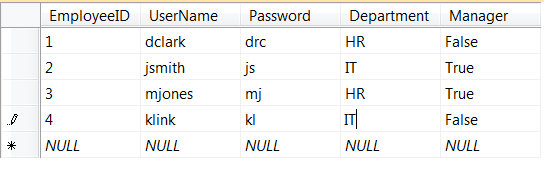
Task – Create a 1:N relationship between Supplier(PK) and Product(FK)

Task – Create a 1:N relationship between Category(PK) and Product(FK)

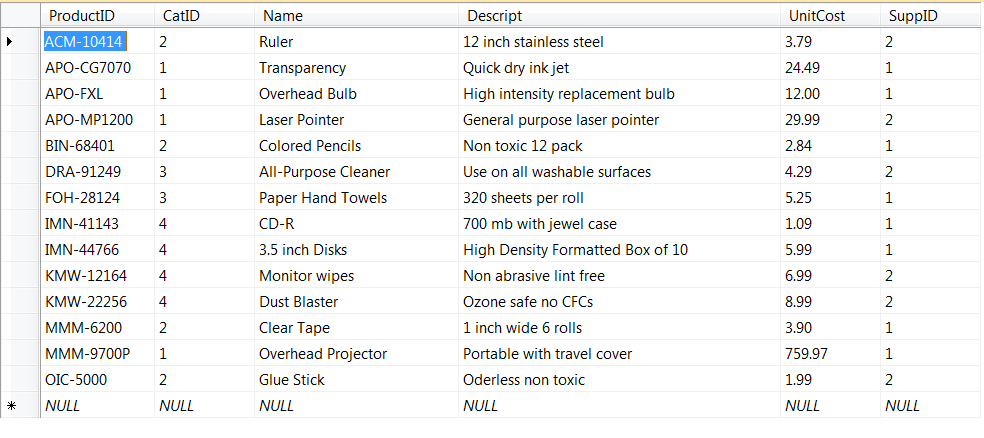
# Performing SQL Queries

Objective: In this section you will be querying and performing CRUD operations on the OfficeSupply database using various DML and SQL statements \*Before you begin performing queries against your database, enter into your tables, the following data exactly as shown in the images:

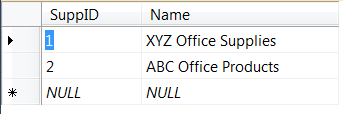
**Employee Table**



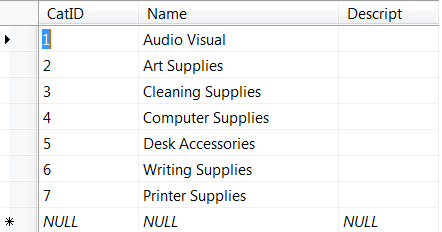
**Product Table**



**Supplier Table**



**Category Table**



## 3.1 SELECT

Task – Select all the rows from the employees table

Task – Select all the rows from the employees table where the Department is HR

Task – Select all the rows from the employees table where username is jsmith and department is HR

Task – Select all the rows from the employees table where manager is true or department is HR

## 3.2 ORDER BY

Task – Select name from product table and order by name in ascending order.

Task – Select name from product table and order by name in descending order.

Task- Select all records from category table order by name.

## 3.3 INSERT INTO

Task – Insert a new row into the employees table.

Task – Insert into a new row into the category table

Task – Insert three records into the supplier table.

## 3.4 UPDATE

Task – Update unit cost in products table where name is ruler

Task – Update the description of computer and cleaning supplies in the Category table.

## 3.5 LIKE

Task – Select username from employees table where username is like “j”

Task – Select name from product table where name is like “O”

## 3.6 BETWEEN

Task – Select name from products table where unitprice is between 3 and 10

Task – Select name from products table where unit price is between 500 and 800

## 3.7 DELETE

Task – Delete a record from the category where the value is audio visual

Task – Delete the three records you previously inserted into the supplier table

# SQL Functions

In this section your will be using the system defined functions built into Oracle 10g as well as creating your own user defined functions

## 4.1 System Defined Scalar Functions

Task – create a function that returns the length of the string of the description of the laser pointer

Task – Create a function the converts a username in the employees table to upper case.

## 4.2 System Defined Aggregate Function

Task – Create a function that gets the sum of the unitprice column from the products table

Task – Create a function that gets the count of all the products in the products table

## 4.3 User Defined Scalar Functions

Task – Create a function that takes two inputs (unit price of products) and calculates the cost of the two products

## 4.4 User Defined Table Valued Functions

Task – Create a function that returns whether or not a username belongs to a manager

# Stored Procedures

In this section you will be creating stored procedures, including stored procedures that have input and output parameters, and return values.

## 5.1 Basic Stored Procedure

Task – Create a store procedure that returns all employees with the username, dept, and manager columns from the employees table. Call the procedure to get the result set.

Task – Create a stored procedure that returns all the products with the name, and unitprice column from the products table.

## 5.2 Stored Procedure Input Parameters

Task – Create a stored procedure that takes in a productID and gets the name and description of that productID

Task – Create a stored procedure that insert a new manager into the employees table

## 5.3 Stored Procedure Output Parameters

Task – Create a stored procedure that calculates the value of the unit cost column in the products table and returns the total amount

Task – Create a procedure that would return username and password based on employeeID

# Transactions

In this section you will be working with transactions. Transactions are usually nested within a stored procedure and contain exaction handling functionality. You will also work with error handling in a transaction.

Task- Create a transaction that is nested inside a stored procedure that inserts a new record into the employees table.

Task – Create a transaction that is nested inside a stored procedure that updates the untitprice of a product in the products table.

Task – create a multi-statement transaction nested in a stored procedure that updates at least two records’ name and description in the category table

# Triggers

In this section you will be creating triggers on various tables. You will work with AFTER/FOR triggers and INSTEAD OF triggers.

## 7.1 AFTER/FOR Triggers

Task - Create an after insert trigger on the categories table fired after a new record is inserted into the table.

Task – Create an after update trigger on the categories table that fires after a row is inserted in the table

Task – Create an after delete trigger on the categories table that fires after a row is deleted from the table.

## 7.2 INSTEAD OF Triggers

Task – create an instead of delete trigger on the Products table that restricts the deletion of any records that are priced below 500 dollars.

# JOINS

In this section you will be performing joins on various tables. You will to populate each table with data. Pay attention to keys and referential integrity when inserting data into your tables.

## 8.1 INNER JOIN

Task – Perform an inner join on tables product and category

Task – Perform an inner join on tables employee and orders

## 8.2 OUTER JOIN

Task – Perform an outer join on tables products and orderitems

Task – Perform an outer join on tables employee and orders

## 8.3 RIGHT JOIN

Task – Perform a right join on tables orders and orderitems

Task – Perform a left join on tables product and orderitems

## 8.4 LEFT JOIN

Task – Perform a left join on tables product and category

Task – Perform a left join on tables employees and orders

## 8.5 CROSS JOIN

Task – Perform a cross join on tables product and category

## 8.6 SELF-JOIN

Task – using the employees table perform a self-join. You can break up the table as needed.

# Views

View can be thought of as virtual tables. In this section you will create views to enhance the security of your database.

Task – create two new columns named SSN and salary on the employees table. Create a view that displays all columns except SSN and salary

Task – Create a view on the products table that only displays only the name of the product and the description.

# Indexes

In this section you will be creating Indexes on various tables. Indexes can speed up performance of reading data.

## 10.1 Clustered Indexes

Task – Create a clustered index on of table of your choice

# Administration

In this section you will be creating backup files of your database. After you create the backup file you will also restore the database.

Task – Create a .bak file for the OfficeSupply database