



­

Hands-on Exercise:

Sub-queries

Version: Understanding ANSI SQL

Table of Contents

[Session 8: Sub-queries 3](#_Toc356566450)

[Exercise 1 3](#_Toc356566451)

[Exercise 2 3](#_Toc356566452)

[Exercise 3 3](#_Toc356566453)

[Exercise 4 4](#_Toc356566454)

[Exercise 5 4](#_Toc356566455)

[Exercise 6 5](#_Toc356566456)

[Exercise 7 5](#_Toc356566457)

Session 8: Sub-queries

# Exercise 1

|  |  |
| --- | --- |
|  |  |
|  |  |
| Hands-on Exercise Objective |
| After completing the hands-on exercises, you will be able to:  Use the different types of sub-queries. |
|  |  |

Problem Statement:

Write a query to display trainer IDs for all associates whose names contain the letter 'i'. Note: Use sub-query to achieve this.

Deliverables Expected:

Records are displayed based on the condition specified.

=>SELECT trainer\_id,associate\_id FROM associate\_status WHERE associate\_id IN

(SELECT associate\_id FROM associate\_info WHERE associate\_name IN

(SELECT associate\_name FROM associate\_info WHERE associate\_name

LIKE '%i%') );

EXERCISE 2

Problem Statement:

Write a query to display trainer IDs for all associates whose names do not contain the letter 'i'.

Note: Use sub-query to achieve this.

Deliverables Expected:

Records are displayed based on the condition specified.

=>SELECT trainer\_id,associate\_id FROM associate\_status WHERE associate\_id IN

(SELECT associate\_id FROM associate\_info WHERE associate\_name IN

(SELECT associate\_name FROM associate\_info WHERE associate\_name

NOT LIKE '%i%') );

# Exercise 3

Problem Statement:

Write a query to display trainer ID and trainer rating for the trainers who handled J2EE module. Hint: Use trainer\_feedback and associate status tables.

Note: Use sub-query to achieve this.

=>SELECT trainer\_id, trainer\_rating from Trainer\_feedback where trainer\_id=

(SELECT trainer\_id from associate\_status where module\_id='J2EE') ;

# Exercise 4

Problem Statement:

Copy all records of a table into another table using sub-query.

**Prerequisite:** Create a new table trainer\_info\_sabbatical with similar structure as that of trainer\_info. Insert few records into the newly created table (to copy these records into trainer\_info table using subquery)

-- create the below table

CREATE TABLE Trainer\_Info\_Sabbatical

(

Trainer\_Id VARCHAR(20) PRIMARY KEY,

Salutation VARCHAR(7) NOT NULL,

Trainer\_Name VARCHAR(30) NOT NULL,

Trainer\_Location VARCHAR(30) NOT NULL,

Trainer\_Track VARCHAR(15) NOT NULL,

Trainer\_Qualification VARCHAR(100) NOT NULL,

Trainer\_Experiance int ,

Trainer\_Email VARCHAR(100) NOT NULL,

Trainer\_Password VARCHAR(20) NOT NULL

);

-- insert the below records

insert into Trainer\_Info\_Sabbatical values ('F011','Mr.','Shyju K', 'Kochi','Java','Bachelor of Technology',9,'shyju@alliance.com','fac11@123');

insert into Trainer\_Info\_Sabbatical values ('F012','Mr.','Raviraj Kumar', 'Kochi','Java','Bachelor of Technology',8,'raviarajkumar@alliance.com','fac12@123');

insert into Trainer\_Info\_Sabbatical values ('F013','Mr.','Suresh Babu N', 'Mumbai','Testing','Bachelor of Technology',19,'sureshbabun@alliance.com','fac13@123');

Deliverables Expected:

Records are displayed based on the condition specified.

# =>CREATE TABLE Trainer\_Info\_Sabbatical

# (

# Trainer\_Id VARCHAR(20) PRIMARY KEY,

# Salutation VARCHAR(7) NOT NULL,

# Trainer\_Name VARCHAR(30) NOT NULL,

# Trainer\_Location VARCHAR(30) NOT NULL,

# Trainer\_Track VARCHAR(15) NOT NULL,

# Trainer\_Qualification VARCHAR(100) NOT NULL,

# Trainer\_Experience int ,

# Trainer\_Email VARCHAR(100) NOT NULL,

# Trainer\_Password VARCHAR(20) NOT NULL

# );

# insert into Trainer\_Info\_Sabbatical values ('F011','Mr.','Shyju K', 'Kochi','Java',

# 'Bachelor of Technology','9','shyju@alliance.com','fac11@123');

# insert into Trainer\_Info\_Sabbatical values ('F012','Mr.','Raviraj Kumar', 'Kochi','Java','Bachelor of Technology','8',

# 'raviarajkumar@alliance.com','fac12@123');

# insert into Trainer\_Info\_Sabbatical values ('F013','Mr.','Suresh Babu N', 'Mumbai','Testing',

# 'Bachelor of Technology','19','sureshbabun@alliance.com','fac13@123');

# INSERT INTO Trainer\_info

# SELECT \* FROM Trainer\_Info\_Sabbatical;

# SELECT \* FROM trainer\_info;

# Exercise 5

|  |  |
| --- | --- |
|  |  |

Problem Statement:

Write a query to update trainer locations in trainer\_info table as ‘Kochi’ for all trainers having their experience, figuring in the output of a query which displays all trainer experiences greater than 10 from trainer\_info\_sabbatical table.

=>UPDATE trainer\_info set trainer\_location="Kochi" where trainer\_experience =

(SELECT trainer\_experience from Trainer\_info\_sabbatical where trainer\_experience > 10);

# Exercise 6

|  |  |
| --- | --- |
|  |  |

Problem Statement:

From trainer\_info table, delete all records where trainer\_experiance matches those in trainer\_info\_sabbatical which are greater than 12.

**Prerequisite :** Have atleast one trainer with experience greater than 12 in both tables.

Deliverables Expected:

Records are displayed based on the condition specified.

=> DELETE FROM trainer\_info where trainer\_experience =

(SELECT trainer\_experience from Trainer\_info\_sabbatical where trainer\_experience > 12);

# Exercise 7

|  |  |
| --- | --- |
|  |  |

Problem Statement:

Using a corelated subquery display all records from trainer\_info\_sabbatical table which also exist in trainer\_info table

**Prerequisite:** insert atleast one recored into trainer\_info\_sabbatical which is not inserted into trainer\_info table and some records which are common to both.

Deliverables Expected:

Records are displayed based on the condition specified.