# Data Engineering Stream Specific Training

# Week – 1 SQL

19-08-2024

Session-1 (Forenoon)

create database collage;

use collage;

create table tblGender

(Id int not null primary key,

Gender nvarchar(15))

insert into tblGender(Id, Gender)

values (1, 'Male'),

(2, 'Female'),

(3, 'others')

insert into tbl\_person(Id, Name, Email, GenderID)

values (1, 'Tom', 'Tom@gmail.com',1),

(2,'Nancy','nancy@gmail.com',2),

(3,'steve','steveh@gmail.com',1),

(4,'robin','robin@gmail.com',3),

(5,'jane','janeh@gmail.com',2)

update tbl\_person set Email='TomUpdated@gmail.com' where Id=1;

update tbl\_person set GenderID=2 where Name='robin';

delete from tbl\_person where GenderID=2;

select \* from tbl\_person;

Session-2(Afternoon)

use collage;

create table Employees(

EmployeeId int primary key identity(1,1),

FirstName varchar(50) not null,

LastName VARCHAR(50) NOT NULL,

Position VARCHAR(50) NOT NULL,

Department VARCHAR(50) NOT NULL,

HireDate DATE NOT NULL

);

INSERT INTO Employees (FirstName, LastName, Position, Department, HireDate)

VALUES ('Steve', 'Harrington', 'Software Engineer', 'IT', '2024-08-19'),

('Jonathan','Byers','Graduate Trainee','Technical','2021-10-11'),

('Eleven','Hopper','Soft skills trainer','soft skills','2018-10-19');

SELECT \* FROM Employees;

SELECT \* FROM Employees WHERE EmployeeID = 1;

UPDATE Employees

SET Position = 'Data Engineer'

WHERE EmployeeID = 2;

DELETE FROM Employees;

INSERT INTO Employees (FirstName, LastName, Position, Department, HireDate)

VALUES

('Amit', 'Sharma', 'Software Engineer', 'IT', '2022-01-15'),

('Priya', 'Mehta', 'Project Manager', 'Operations', '2023-02-20'),

('Raj', 'Patel', 'Business Analyst', 'Finance', '2021-06-30'),

('Sunita', 'Verma', 'HR Specialist', 'HR', '2019-08-12'),

('Vikram', 'Rao', 'Software Engineer', 'IT', '2021-03-18'),

('Anjali', 'Nair', 'HR Manager', 'HR', '2020-05-14'),

('Rohan', 'Desai', 'Finance Manager', 'Finance', '2022-11-25'),

('Sneha', 'Kumar', 'Operations Coordinator', 'Operations', '2023-07-02'),

('Deepak', 'Singh', 'Data Scientist', 'IT', '2022-08-05'),

('Neha', 'Gupta', 'Business Analyst', 'Finance', '2020-10-10');

-- 3 Variations

select FirstName,LastName,Department from Employees;

select \* from Employees where Department='IT';

select \* from Employees where HireDate>'2022-01-01';

-- Queries

select \* from Employees where Department in ('HR','IT');

select distinct Department from Employees

select \* from employees where Department='IT' and HireDate>'2022-01-01'

select \* from employees where Department='IT' or HireDate>'2022-01-01'

-- Queries

select \* from Employees where HireDate between '2022-01-01' and '2022-12-31';

select \* from Employees where LastName like 'S%'

select FirstName+' '+LastName as FullName, Department from Employees;

-- Table Alias

select E.FirstName, E.LastName, E.Department

from Employees as E

where E.Department='IT';

-- Queries

select count(\*) as EmployeeCount From Employees;

select Department, count(\*) As Employeecount from Employees group by Department;

create table Departments(DeptID int primary key,

DeptName Varchar(50)

);

insert into Departments(DeptID, DeptName)

values(1,'IT'),

(2,'HR'),

(3,'Finance'),

(4,'Operations');

-- Join

select e.employeeId, e.firstname,e.lastname, e.position,d.deptname

from employees e join Departments d on e.Department=d.DeptName;

-- Sub Query

select FirstName, LastName from Employees

where HireDate=(select min(HireDate) from Employees);

SELECT FirstName, LastName

FROM Employees

WHERE Department IN (

SELECT Department

FROM Employees

GROUP BY Department

HAVING COUNT(\*) > 2

);

20-08-2024

Day-2(Forenoon)

create database companydb;

use companydb;

create table tblEmployee(

ID int primary key,

Name nvarchar(50),

Gender nvarchar(10),

salary int,

DepartmentId int

);

create table tblDepartment(

ID int primary key,

DepartmentName nvarchar(50),

Location nvarchar(50),

DepartmentHead nvarchar(50),

);

Insert into tblEmployee(ID, Name, Gender, salary, DepartmentId)

values

(1, 'Tom','Male',4000, 1),

(2, 'Pam', 'Female', 3000, 3),

(3, 'John', 'Male', 3500, 1),

(4, 'Sam', 'Male', 4500, 2),

(5, 'Todd', 'Male', 2800, 2),

(6, 'Ben', 'Male', 7000, 1),

(7, 'Sara', 'Female', 4800, 3),

(8, 'Valarie', 'Female', 5500, 1),

(9, 'James', 'Male', 6500, NULL),

(10, 'Russell', 'Male', 8800, NULL);

Insert into tblDepartment(ID, DepartmentName, Location, DepartmentHead)

values

(1, 'IT', 'London','Rick'),

(2, 'Payroll', 'Delhi', 'Ron'),

(3, 'HR', 'New York', 'Christie'),

(4, 'Other Department', 'Sydney', 'Cindrella');

-- Inner Join

select name, gender, salary, departmentname from tblEmployee

inner join tblDepartment

on tblEmployee.DepartmentId=tblDepartment.ID

-- Left outer join/ Left join

select name, gender, salary, departmentname from tblEmployee

left outer join tblDepartment

on tblEmployee.DepartmentId=tblDepartment.ID

-- Right outer join/Right Join

select name, gender, salary, departmentname from tblEmployee

right outer join tblDepartment

on tblEmployee.DepartmentId=tblDepartment.ID

-- Full outer join

select name, gender, salary, departmentname from tblEmployee

full outer join tblDepartment

on tblEmployee.DepartmentId=tblDepartment.ID

create table Products(

product\_id int primary key,

product\_name nvarchar(50),

price decimal(10,2)

);

create table Orders(

order\_id int primary key,

product\_id int,

quantity int,

order\_date Date

);

Insert into Products(product\_id,product\_name,price)

values

(1,'Laptop',800.00),

(2,'SmartPhone',500.00),

(3, 'Tablet', 300.00),

(4, 'Headphones', 50.00),

(5, 'Monitor', 150.00);

Insert into Orders (order\_id, product\_id, quantity, order\_date)

values

(1, 1, 2, '2024-08-01'),

(2, 2, 1, '2024-08-02'),

(3, 3, 3, '2024-08-03'),

(4, 1, 1, '2024-08-04'),

(5, 4, 4, '2024-08-05'),

(6, 5, 2, '2024-08-06'),

(7, 6, 1, '2024-08-07');

-- inner join

select order\_id, product\_name, quantity, order\_date

from Orders

inner join Products

on Orders.product\_id=Products.product\_id;

-- Left Outer Join

select order\_id, product\_name, quantity, order\_date

from Orders

left outer join Products on Orders.product\_id = Products.product\_id;

-- Right Outer Join

select order\_id, product\_name, quantity, order\_date

from Products

right outer join Orders on Orders.product\_id = Products.product\_id;

-- Full outer join

select order\_id, product\_name, quantity, order\_date

from Orders

full outer join Products on Orders.product\_id = Products.product\_id;

-- Grouping Set

select p.product\_name, o.order\_date, sum(o.quantity)

as total\_quantity from Orders o

join products p on o.product\_id=p.product\_id

group by GROUPING sets ((p.product\_name),(o.order\_date))

-- Sub Query

select o.order\_id, o.product\_id,

(select p.product\_name from Products p where p.product\_id=o.product\_id) as product\_name

from Orders o

-- Where clause sub query

select order\_id, order\_date, product\_id

from orders

where product\_id in (select product\_id from Products where price>500);

-- Exists

select u.user\_id, u.user\_name from Users u

where exists(select 1 from Orders o where o.user\_id=u.user\_id);

-- Any

select p.product\_name, p.price

from Products p

where p.price> any(select price from Products where product\_name like 'LAPTOP%');

-- ALL

select p.product\_name, p.price

from Products p

where p.price> All(select price from Products where product\_name like 'smartphone%');

-- Nested Sub Query

select user\_id, user\_name

from Users

where user\_id in (

select user\_id

from Orders

where product\_id in (

select product\_id

from Products

where price>1000

)

);

-- Union

select product\_name from Products where price>500

union

select product\_name from Products where product\_name like 'smart%'

-- Intersection

select product\_name from Products where price>500

Intersect

select product\_name from Products where product\_name like 'smart%'

-- Except

select product\_name from products where price>500

Except

select product\_name from Products where product\_name like 'smart%'

Day-2(Afternoon)

use companydb

CREATE TABLE Employees (

employee\_id INT PRIMARY KEY,

employee\_name VARCHAR(255),

department VARCHAR(255),

manager\_id INT

);

CREATE TABLE Salaries (

salary\_id INT PRIMARY KEY,

employee\_id INT,

salary DECIMAL(10, 2),

salary\_date DATE,

FOREIGN KEY (employee\_id) REFERENCES Employees(employee\_id)

);

INSERT INTO Employees (employee\_id, employee\_name, department, manager\_id) VALUES

(1, 'John Doe', 'HR', NULL),

(2, 'Jane Smith', 'Finance', 1),

(3, 'Robert Brown', 'Finance', 1),

(4, 'Emily Davis', 'Engineering', 2),

(5, 'Michael Johnson', 'Engineering', 2);

INSERT INTO Salaries (salary\_id, employee\_id, salary, salary\_date) VALUES

(1, 1, 5000, '2024-01-01'),

(2, 2, 6000, '2024-01-15'),

(3, 3, 5500, '2024-02-01'),

(4, 4, 7000, '2024-02-15'),

(5, 5, 7500, '2024-03-01');

-- Equi Join

select employee\_name, department, salary, salary\_date

from Employees

join Salaries on Employees.employee\_id = Salaries.employee\_id;

-- Self Join

select e1.employee\_name as Employee, e2.employee\_name as manager from Employees e1

left join Employees e2 on e1.manager\_id=e2.employee\_id

-- Group by with Having

select e.department, AVG(s.salary) as average\_salary

from Employees e

join Salaries s on e.employee\_id = s.employee\_id

group by e.department

having avg(S.salary) >= 6000;

-- Group by with grouping sets

select e.department, sum(s.salary) as total\_salary

from Employees e

join salaries s on e.employee\_id = s.employee\_id

group by grouping sets((e.department),());

-- SubQuery

select e.employee\_name, s.salary

from Employees e

join Salaries s on e.employee\_id = s.employee\_id

where S.salary > (select avg(salary) from Salaries);

-- Exists

select E.employee\_name

from Employees E

where exists (

select 1 from Salaries S where S.employee\_id = E.employee\_id

and S.salary\_date >= '2024-01-01'

and S.salary\_date < '2025-01-01'

);

-- Any

select e.employee\_name, s.salary

from Employees e

join Salaries s on e.employee\_id = s.employee\_id

where s.salary > any (

select s2.salary

from Employees e2

join Salaries s2 on e2.employee\_id = s2.employee\_id

where e2.department = 'Engineering'

);

-- All

select e.employee\_name, s.salary

from Employees e

join Salaries s on e.employee\_id = s.employee\_id

where s.salary > all (

select s2.salary

from Employees e2

join Salaries s2 on e2.employee\_id = s2.employee\_id

where e2.department = 'Finance'

);

-- Nested SubQueries

select e.employee\_name, s.salary

from employees e

join salaries s on e.employee\_id = s.employee\_id

where s.salary > (

select avg(s2.salary)

from salaries s2

where s2.employee\_id in (

select e2.employee\_id

from employees e2

where e2.department = 'hr'

)

);

-- Correlated SubQueries

select e.employee\_name, s.salary

from employees e

join salaries s on e.employee\_id = s.employee\_id

where s.salary > (

select avg(s2.salary)

from employees e2

join salaries s2 on e2.employee\_id = s2.employee\_id

where e2.department = e.department

);

-- Union

select employee\_name from employees

where department = 'hr'

union

select employee\_name from employees

where department = 'finance';

-- Intersect

select employee\_name from employees

where department = 'finance'

intersect

select employee\_name from employees

where department = 'engineering';

-- Except

select employee\_name from employees

where department = 'finance'

except

select employee\_name from employees

where department = 'hr';

-- Merge

create table salaryrevisions (

employee\_id int,

new\_salary decimal(10, 2)

);

insert into salaryrevisions (employee\_id, new\_salary) values

(1, 5500),

(2, 6200),

(6, 7800);

merge into salaries as target

using salaryrevisions as source

on target.employee\_id = source.employee\_id

when matched then

update set target.salary = source.new\_salary

when not matched by target then

insert (salary\_id, employee\_id, salary, salary\_date)

values ((select isnull(max(salary\_id), 0) + 1 from salaries),

source.employee\_id,

source.new\_salary, getdate());

Day-3

create procedure GetAllProducts

as

begin

select \* from Courses;

end

exec GetAllProducts;

create procedure GetProductId

@productId int

as

begin

select \* from products

where productId=@productId

end;

exec GetProductId @productId=1;