**SQL and PL SQL**

**Create Tables and Insert &Update Values**

## **Joins,  SubQueries, Packages(Packages Spec &Packages body) and Functions**

# **Table name**

* **Fish Species**
* **Fishing Activity**
* **Marine Habitats**

**Fish Species**

CREATE TABLE FishSpecies (

SpeciesID NUMBER PRIMARY KEY,

SpeciesName VARCHAR2(50) NOT NULL,

Population NUMBER,

);

INSERT INTO FishSpecies (SpeciesID, SpeciesName, Population)

VALUES (1, 'kumar', 10000),

(2, 'vijay’, 5000),

(3, 'Ajith', 800)

;

SELECT \* FROM FishSpecies;

UPDATE FishSpecies

SET Population = Population - 100

WHERE SpeciesID = 1;

DELETE FROM FishSpecies

WHERE SpeciesID = 2;

**FishingActivity**

CREATE TABLE FishingActivity (

FishingActivityID NUMBER PRIMARY KEY,

Date TIMESTAMP,

Location VARCHAR2(100),

CatchQuantity NUMBER,

SpeciesID NUMBER,

);

INSERT INTO FishingActivity (ActivityID, Date, Location, CatchQuantity, SpeciesID)

VALUES (1, TO\_TIMESTAMP('0001', 'YYYY-MM-DD HH24:MI:SS'), 'Pacific Ocean', 100, 1),

(2, TO\_TIMESTAMP('0001', 'YYYY-MM-DD HH24:MI:SS'), 'Atlantic Ocean', 200, 2);

SELECT \* FROM FishingActivity;

DELETE FROM FishingActivity

WHERE ActivityID = 2;

SELECT \* FROM FishingActivity

WHERE CatchQuantity > 100;

**MarineHabitats**

CREATE TABLE MarineHabitats (

MarineHabitatsID NUMBER PRIMARY KEY,

HabitatType VARCHAR2(50),

Location VARCHAR2(100),

ConservationStatus VARCHAR2(50) );

INSERT INTO MarineHabitats (MarineHabitatsID, HabitatType, Location, ConservationStatus)

VALUES (1, 'Coral Reef', 'Great Barrier Reef', 'Protected'),

(2, 'Mangrove Forest', 'Amazon Delta', 'Endangered');

SELECT \* FROM MarineHabitats

DELETE FROM MarineHabitats

WHERE ActivityID = 2;

SELECT \* FROM MarineHabitats

WHERE Location = 'Pacific Ocean';

SELECT \* FROM MarineHabitats

WHERE ConservationStatus = 'Protected';

**Sub Quries**

SELECT SpeciesName, SUM(CatchQuantity) AS TotalCatch

FROM FishingActivity

JOIN FishSpecies ON FishingActivity.SpeciesID = FishSpecies.SpeciesID

GROUP BY SpeciesName;

SELECT \*

FROM FishingActivity

WHERE SpeciesID = (SELECT SpeciesID FROM FishSpecies WHERE SpeciesName = 'Tuna');

SELECT Location, (SELECT SUM(CatchQuantity) FROM FishingActivity WHERE Location = a.Location) AS TotalCatch

FROM (SELECT DISTINCT Location FROM FishingActivity) a ;

SELECT \*

FROM MarineHabitats

WHERE HabitatID IN (SELECT HabitatID FROM FishSpecies WHERE Category = 'Edible');

SELECT \*

FROM FishSpecies

WHERE FishID IN (SELECT FishID FROM FishingActivity WHERE Location = 'Atlantic Ocean');

SELECT \*

FROM FishingActivity

WHERE HabitatID IN (SELECT HabitatID FROM MarineHabitats WHERE ProtectionStatus != 'Protected');

**Joins**

SELECT FishSpecies.SpeciesName, FishingActivity.Date, FishingActivity.CatchQuantity

FROM FishSpecies

INNER JOIN FishingActivity ON FishSpecies.SpeciesID = FishingActivity.SpeciesID;

SELECT \*

FROM FishSpecies

CROSS JOIN FishingActivity;

SELECT \*

FROM FishSpecies

INNER JOIN FishingActivity ON FishSpecies.HabitatID = FishingActivity.HabitatID;

SELECT \*

FROM FishSpecies

LEFT JOIN FishingActivity ON FishSpecies.HabitatID = FishingActivity.HabitatID;

SELECT \*

FROM FishSpecies

RIGHT JOIN FishingActivity ON FishSpecies.HabitatID = FishingActivity.HabitatID;

SELECT \*

FROM FishSpecies

FULL OUTER JOIN FishingActivity ON FishSpecies.HabitatID = FishingActivity.HabitatID;

# **packages**

CREATE OR REPLACE PACKAGE MarineManagementPackage AS

PROCEDURE InsertFishSpecies(p\_SpeciesName VARCHAR2, p\_Category VARCHAR2);

PROCEDURE UpdateFishingActivityLocation(p\_FishingActivityID NUMBER, p\_NewLocation VARCHAR2);

FUNCTION GetProtectedHabitats RETURN SYS\_REFCURSOR;

END MarineManagementPackage;

CREATE OR REPLACE PACKAGE BODY MarineManagementPackage AS

PROCEDURE InsertFishSpecies(p\_SpeciesName VARCHAR2, p\_Category VARCHAR2) IS

BEGIN

-- Your logic to insert a new fish species into the FishSpecies table

END InsertFishSpecies;

PROCEDURE UpdateFishingActivityLocation(p\_FishingActivityID NUMBER, p\_NewLocation VARCHAR2) IS

BEGIN

-- Your logic to update the location of a fishing activity in the FishingActivity table

END UpdateFishingActivityLocation;

FUNCTION GetProtectedHabitats RETURN SYS\_REFCURSOR IS

v\_Cursor SYS\_REFCURSOR;

BEGIN

OPEN v\_Cursor FOR

SELECT \*

FROM MarineHabitats

WHERE ProtectionStatus = 'Protected';

RETURN v\_Cursor;

END GetProtectedHabitats;

END MarineManagementPackage;

**PL/SQL procedure using CURSOR**

CREATE OR REPLACE PROCEDURE DisplayFishSpecies IS

v\_Cursor SYS\_REFCURSOR;

v\_SpeciesName FishSpecies.SpeciesName%TYPE;

BEGIN

OPEN v\_Cursor FOR

SELECT SpeciesName FROM FishSpecies;

LOOP

FETCH v\_Cursor INTO v\_SpeciesName;

EXIT WHEN v\_Cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Fish Species: ' || v\_SpeciesName);

END LOOP;

CLOSE v\_Cursor;

END DisplayFishSpecies;

-- Example of a PL/SQL function with EXCEPTIONS

CREATE OR REPLACE FUNCTION GetTotalCatch(p\_Year NUMBER) RETURN NUMBER IS

v\_TotalCatch NUMBER;

BEGIN

SELECT SUM(CatchQuantity) INTO v\_TotalCatch

FROM FishingActivity

WHERE Year = p\_Year;

RETURN v\_TotalCatch;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No data found for the given year.');

RETURN NULL;

END GetTotalCatch;

**PL/SQL collection (Nested Table)**

DECLARE

TYPE SpeciesList IS TABLE OF FishSpecies.SpeciesName%TYPE;

v\_FishSpeciesList SpeciesList;

BEGIN

SELECT SpeciesName BULK COLLECT INTO v\_FishSpeciesList

FROM FishSpecies;

FOR i IN 1..v\_FishSpeciesList.COUNT LOOP

DBMS\_OUTPUT.PUT\_LINE('Fish Species: ' || v\_FishSpeciesList(i));

END LOOP;

END;

**PL/SQL record**

DECLARE

TYPE FishingActivityRecord IS RECORD (

ActivityID NUMBER,

Location VARCHAR2(100)

);

v\_ActivityInfo FishingActivityRecord;

BEGIN

SELECT ActivityID, Location INTO v\_ActivityInfo

FROM FishingActivity

WHERE ROWNUM = 1;

DBMS\_OUTPUT.PUT\_LINE('Activity ID: ' || v\_ActivityInfo.ActivityID);

DBMS\_OUTPUT.PUT\_LINE('Location: ' || v\_ActivityInfo.Location);

END;

**index in a PL/SQL query**

DECLARE

v\_FishID FishSpecies.FishID%TYPE;

BEGIN

SELECT FishID INTO v\_FishID

FROM FishSpecies

WHERE SpeciesName = 'Tuna';

END;

**PL/SQL BULK COLLECT and FORALL for updating locations in FishingActivity**

DECLARE

TYPE LocationList IS TABLE OF FishingActivity.Location%TYPE;

v\_NewLocations LocationList;

BEGIN

-- ---Populate the new locations

v\_NewLocations := LocationList('New Location 1', 'New Location 2', 'New Location 3');

-- ---Update locations using FORALL

FORALL i IN 1..v\_NewLocations.COUNT

UPDATE FishingActivity

SET Location = v\_NewLocations(i)

WHERE ROWNUM = i;

END;