

## Forest Department Database Management System

Comprehensive SQL Database Solution

**Presented By:** Mansi Shitole

**Course:** Java Full Stack | **Mentor:** Shalini Verma Ma'am



Project Introduction





Comprehensive System

25 interconnected tables covering all forest operations and wildlife management.



Real-World Impact

Practical implementation for wildlife conservation and government forest departments.



Scalable Solution

End-to-end SQL implementation with advanced features for future expansion.

## Why Forest Management?

Environmental Impact



* Critical for biodiversity

conservation

* Essential for climate change

mitigation

* Supports sustainable resource management

Operational Benefits

* Enables effective wildlife

protection

* Facilitates scientific research

and monitoring

* Improves disaster management capabilities

Project Scope & Objectives



1

Centralised Data Management

Unified forest data repository with real-time monitoring of wildlife populations.

2

Resource Optimisation

Efficient resource allocation and automated reporting system for operations.

3

Enhanced Conservation

Multi-department integration with advanced analytics for conservation efforts.

4

Scalable Architecture

User-friendly interface design with expandable system capabilities.

## Technology Implementation

Database Foundation

MySQL with 25 normalised tables, advanced SQL features, and comprehensive constraints.

Core Components

DDL, DML, DQL operations with stored procedures, functions, views, and triggers.

Advanced Features

Transaction management and robust security implementation for data protection.





## Phase 1: Database Foundation

Building a comprehensive database system for forest department operations, wildlife conservation, and resource management.

Database Design

25 interconnected tables with comprehensive constraints and ER diagram design.

Data Population

500+ sample records with realistic data and referential integrity.

Query Implementation

Basic CRUD operations for forest zone and wildlife management.

Database Schema Architecture

1

2

3

Forest Zones & Boundaries

Zone management and geographical

tracking.

Wildlife Management

Biodiversity tracking and species

monitoring.

Staff & Operations

Human resources and duty allocation.

4

5

6

Research & Conservation

Scientific studies and conservation

initiatives.

Tourism & Public Engagement

Visitor management and public

programmes.

Equipment & Infrastructure

Asset tracking and facility management.

7

Budget & Finance

Financial planning and resource allocation.





## Table Creation Example

Sample SQL structure for the ForestZones table, demonstrating proper

schema design with data types and constraints.

-- Sample Table CreationCREATE TABLE ForestZones ( id INT

PRIMARY KEY, name VARCHAR(100), location VARCHAR(100),

area\_sq\_km FLOAT, zone\_type VARCHAR(50), established\_date DATE, status VARCHAR(20));



Advanced Constraints Implementation

preencoded.png

Primary Keys

Unique identifiers ensuring each record can be distinctly referenced across all tables.

preencoded.png

Foreign Keys

Inter-table relationships maintaining referential integrity throughout the database.

preencoded.png

Data Validation

NOT NULL for mandatory fields, CHECK constraints for validation, UNIQUE for duplicate

prevention.

preencoded.png

Automation

DEFAULT values for automatic field population and streamlined data entry.



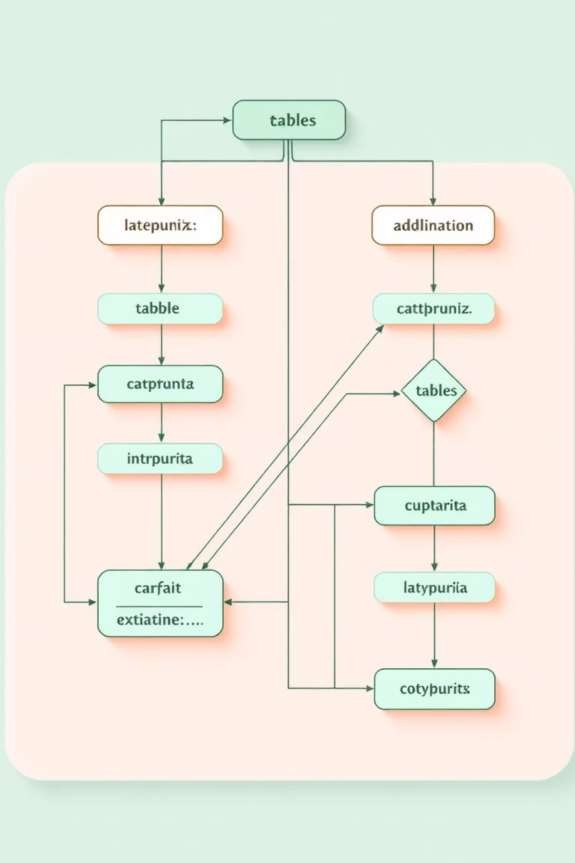
#### preencoded.pngBasic Query Operations

Essential SQL commands for database creation, data manipulation, and

retrieval operations.



-- Database CreationCREATE DATABASE ForestDepartment;-- Table SelectionSELECT \* FROM ForestZones;-- Data InsertionINSERT INTO ForestZones VALUES (1, 'Sundarbans Reserve', 'West Bengal', 4260.0, ...);-- Data RetrievalSELECT name, location FROM ForestZones WHERE status='Active';



Entity-Relationship Diagram

Comprehensive mapping of database relationships ensuring efficient query design and data integrity.

ForestZones

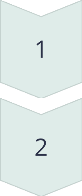
PatrolRecords

Staff

RangerAssignments

WildlifeSpecies

AnimalTracking

ConservationProjects

ResearchActivities

BudgetAllocations

Departments





**Benefits:** Clear relationship mapping, referential integrity assurance, and efficient query design

throughout the system.

## Phase 1 Achievements



Complete Structure

25 tables with proper attributes and relationships.



500+ Records

Realistic sample data across all tables.



Comprehensive Constraints

Full data integrity and validation.



CRUD Operations

Basic query implementation complete.



Documentation

Complete technical documentation.



Real-World Applications

Forest Zone Management

Comprehensive tracking of protected

areas and boundaries.

Wildlife Population Tracking

Monitoring endangered species and

biodiversity.

Staff Duty Allocation

Efficient human resource management

and scheduling.

Incident Reporting System

Real-time event management and

response coordination.

Research Project Monitoring

Tracking scientific studies and

conservation initiatives.

Tourism Management

Visitor coordination and public

engagement programmes.





## Phase 2: Query Mastery

Advanced DDL operations, complex DML queries, comprehensive DQL implementation, and cascade operations.

### DDL Operations



Table Modification

ALTER TABLE Staff to add emergency contact column for enhanced record-keeping.

Constraint Addition

CHECK constraint ensures population estimates remain positive values.

Index Creation

CREATE INDEX on species type improves query performance significantly.

ALTER TABLE Staff ADD COLUMN emergency\_contact VARCHAR(15);ALTER TABLE WildlifeSpecies ADD CONSTRAINT chk\_population CHECK (population\_estimate > 0);CREATE INDEX idx\_species\_type ON WildlifeSpecies(species\_type);

DML Operations

Bulk Updates

10% salary increase for all Research department staff

members.

Conditional Deletion

Remove resolved incidents from before 2024 to maintain

database efficiency.



UPDATE Staff SET salary = salary \* 1.1 WHERE

department = 'Research';

DELETE FROM Incidents WHERE status = 'Resolved'

AND date < '2024-01-01';



## Advanced DQL Queries

Aggregation with Grouping

Count species and calculate average population by

type for comprehensive wildlife analysis.

SELECT species\_type, COUNT(\*) as

species\_count,AVG(population\_estimate) as avg\_populationFROM WildlifeSpecies GROUP BY species\_type;

Complex Filtering

Filter large zones by type and sort by establishment

date for historical insights.

SELECT \* FROM ForestZones WHERE area\_sq\_km

> 1000 AND zone\_type IN ('National Park', 'Tiger Reserve')ORDER BY established\_date DESC;



Operators Implementation

preencoded.png

Comparison Operators

=, <, >, <=, >=, != for value comparisons

preencoded.png

Logical Operators

AND, OR, NOT for combining conditions

preencoded.png

Special Operators

IN, BETWEEN, LIKE, IS NULL for advanced filtering

preencoded.png

Arithmetic Operators

+, -, \*, / for mathematical calculations



Clause Implementation

WHERE with Multiple Conditions

Filter zones between 500-2000 sq km with Active status using BETWEEN operator.

GROUP BY with HAVING

Count staff by department and filter groups with more than 3 members.

ORDER BY Multiple Columns

Sort wildlife species by conservation status, then population estimate descending.



SELECT name, location, area\_sq\_km FROM ForestZones WHERE area\_sq\_km BETWEEN 500 AND 2000 AND status =

'Active';

Phase 2 Achievements

# 20

DDL Queries

Table modifications and constraints

 Mastered all SQL query types DDL, DML, and DQL operations executed successfully

# 20

DML Queries

Data manipulation operations

Implemented complex constraints

CHECK constraints and foreign keys

with cascade operations

# 20

DQL Queries

Complex data retrieval

Used all major operators and clauses

Comparison, logical, special operators with WHERE, GROUP BY, HAVING, ORDER BY





Phase 3: Advanced Data Retrieval

Master join operations, subqueries, and function development for complex data

analysis in forest management systems.



Join Operations

20 join queries mastering data relationships

Subquery Operations

20 advanced subquery implementations

Function Development

20 built-in and user-defined functions

Complex Analysis

Enhanced data retrieval capabilities

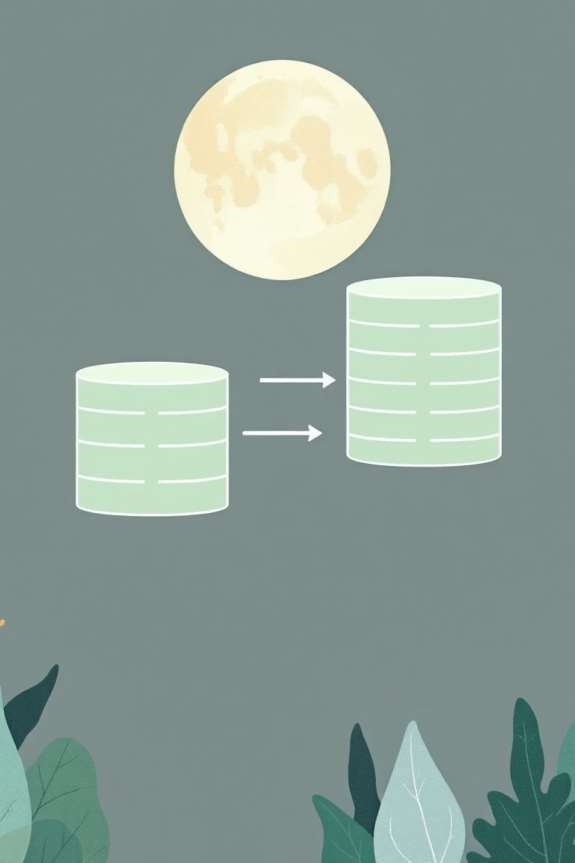
Join Operations Fundamentals

INNER JOIN

Retrieve matching records from forest zones and wildlife species tables.

LEFT JOIN

Comprehensive data including zones without patrol records.



fz.name;

SELECT fz.name, ws.species\_name, ws.conservation\_statusFRO M ForestZones fzINNER JOIN WildlifeSpecies ws ON fz.id = ws.zone\_id;

SELECT fz.name, COUNT(pr.id) as patrol\_countFROM ForestZones fzLEFT JOIN PatrolRecords pr ON fz.id

= pr.zone\_idGROUP BY

#### Subquery Implementation

Subquery with IN

Filter wildlife species based on endangered health status from tracking records.

Correlated Subquery

Compare species population

against type-specific averages.

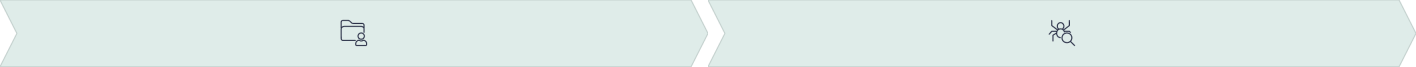
SELECT name, population\_estimateFROM WildlifeSpecies ws1WHERE population\_estimate > ( SELECT

AVG(population\_estimate) FROM WildlifeSpecies ws2 WHERE ws2.species\_type = ws1.species\_type);



SELECT name, species\_typeFROM WildlifeSpeciesWHERE id IN ( SELECT species\_id FROM AnimalTracking WHERE health\_status = 'Endangered');

Complex Subquery Patterns

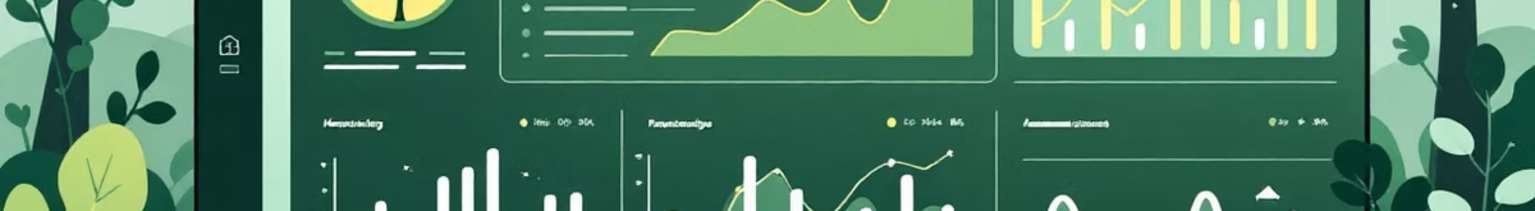


FROM Clause Subquery

Calculate average incidents per zone, filtering high-incident areas.

EXISTS Subquery

Identify forest zones with pending incidents requiring attention.





-- FROM Clause ExampleSELECT zone\_name, avg\_incidentsFROM ( SELECT fz.name AS zone\_name, AVG(pr.incidents\_reported) AS avg\_incidents FROM ForestZones fz JOIN PatrolRecords pr ON fz.id = pr.zone\_id GROUP BY fz.name) AS zone\_statsWHERE avg\_incidents > 2;

Built-in Functions Mastery



String Functions

UPPER, LOWER, and CONCAT for text manipulation.

SELECT UPPER(name) AS species\_name\_upper,

species\_type, ')') FROM WildlifeSpecies;

CONCAT(name, ' (',



Date Functions

YEAR, DATEDIFF, and CURDATE for temporal analysis.

SELECT name, YEAR(established\_date) AS year, DATEDIFF(CURDATE(),

established\_date)FROM ForestZones;

Aggregate Functions in Action

Comprehensive statistical analysis of wildlife populations by species type.

SELECT species\_type,

COUNT(\*) AS total\_species,

MIN(population\_estimate) AS min\_population,

MAX(population\_estimate) AS max\_population, AVG(population\_estimate) AS avg\_population, SUM(population\_estimate) AS total\_populationFROM

WildlifeSpeciesGROUP BY species\_typeHAVING total\_species > 2;

5



Aggregate Functions COUNT, MIN, MAX, AVG, SUM

1

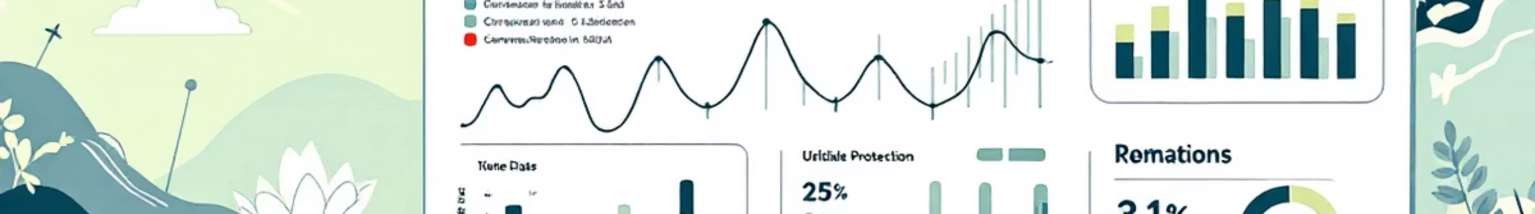
Grouping Clause

GROUP BY species type

1

Filter Condition

HAVING clause applied



User-Defined Functions

Conservation Priority UDF

CREATE FUNCTION GetConservationPriority( population INT, status VARCHAR(50)) RETURNS VARCHAR(20)DETERMINISTICBEGIN IF population < 1000 AND status = 'Endangered' THEN RETURN 'CRITICAL'; ELSEIF population BETWEEN 1000 AND 5000 THEN RETURN 'HIGH'; ELSE RETURN 'MEDIUM'; END IF;END;

Custom function evaluating species priority based on population and status.

* CRITICAL: Population < 1,000 and endangered
* HIGH: Population 1,000–5,000
* MEDIUM: All other cases





Phase 3 Achievements

|  |
| --- |
| Join Operations Mastered  INNER, LEFT, multiple table, and self joins implemented successfully. |
|  |
| Complex Subqueries  IN, correlated, FROM clause, and EXISTS subqueries deployed. |
|  |
| Function Development  Built-in functions utilised; multiple UDFs created for custom logic. |
|  |
| Enhanced Analysis  Improved query performance and data analysis capabilities achieved. |



## Phase 4: Advanced Database Features

Elevating database capabilities through sophisticated features including views, stored procedures, transaction control, triggers, and comprehensive security implementation.

Views Implementation

Create virtual tables for simplified data access and reporting

Stored Procedures

Develop reusable database operations with parameters

Transaction Control

Ensure data integrity with TCL commands

Trigger Automation

Automate responses to database events

Security Implementation

Control access with DCL privileges



###### Views Implementation

Active Conservation Projects

Virtual table displaying ongoing projects with key details including name, objective, dates, budget, and lead scientist. Filters for active status and future end dates.

Wildlife Statistics Summary

Aggregated view providing species counts and average population estimates grouped by species type for quick statistical analysis.

CREATE VIEW WildlifeSummary ASSELECT species\_type, COUNT(\*) as species\_count, AVG(population\_estimate) as avg\_populationFROM WildlifeSpeciesGROUP BY species\_type;



CREATE VIEW ActiveProjects

ASSELECT name, objective,

start\_date,

end\_date,

budget, lead\_scientistFROM

ConservationProjectsWHERE

status = 'Ongoing'

AND

end\_date > CURDATE();



Advanced Stored Procedures

Multi-Operation Procedures

DELIMITER //CREATE PROCEDURE UpdateSpeciesPopulation( IN species\_id INT, IN new\_population INT, OUT old\_population INT, OUT status\_message VARCHAR(100))BEGIN SELECT population\_estimate INTO old\_population FROM WildlifeSpecies WHERE id = species\_id; UPDATE WildlifeSpecies SET population\_estimate = new\_population, updated\_at = NOW() WHERE id = species\_id; SET status\_message = CONCAT( 'Population updated from ', old\_population, ' to ', new\_population

);END //DELIMITER ;

Sophisticated procedures handling multiple database operations with both input and output parameters.

* Retrieve current population value
* Update species population estimate
* Set timestamp for modification
* Return old value and status message



Cursor Implementation

Cursors enable row-by-row processing of query results for complex batch operations requiring conditional logic.



Declare Cursor

Define cursor with SELECT query for species data

Open & Fetch

Open cursor and retrieve rows iteratively



Process Logic

Apply conditional updates based on population thresholds

Close Cursor

Release cursor resources after processing



DELIMITER //CREATE PROCEDURE UpdateAllSpeciesStatus()BEGIN DECLARE done INT DEFAULT FALSE; DECLARE species\_id INT; DECLARE current\_population INT; DECLARE cur CURSOR FOR SELECT id, population\_estimate FROM WildlifeSpecies; DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE; OPEN cur; read\_loop: LOOP FETCH cur INTO species\_id, current\_population; IF done THEN LEAVE read\_loop; END IF; IF current\_population < 1000 THEN UPDATE WildlifeSpecies SET conservation\_status = 'Critical' WHERE id = species\_id; END IF; END LOOP; CLOSE cur;END //DELIMITER ;

Window Functions





Species Population Ranking

RANK() and ROW\_NUMBER() functions partition by species type and order by population, providing both

category-specific and overall rankings.



Moving Averages

Calculate rolling averages for patrol area coverage using window frame specification with 2 preceding rows and current row.

SELECT name, species\_type, population\_estimate, RANK() OVER ( PARTITION BY species\_type ORDER BY population\_estimate DESC ) AS population\_rank, ROW\_NUMBER() OVER ( ORDER BY population\_estimate DESC ) AS overall\_rankFROM WildlifeSpecies;

SELECT date, area\_covered, AVG(area\_covered) OVER ( ORDER BY date ROWS BETWEEN 2 PRECEDING AND CURRENT ROW ) AS

moving\_avg\_areaFROM PatrolRecords;

Transaction Control (TCL)

Ensure data integrity through atomic operations with commit and rollback capabilities for secure budget transfers.

Start Transaction

Initiate atomic operation block

Add Budget

Increase Conservation department allocation by £50,000

Commit/Rollback

Finalise or undo changes based on validation

Deduct Budget

Reduce Research department allocation by £50,000

1

2

3

4

5

Error Check

Verify operation success or trigger rollback



START TRANSACTION;UPDATE BudgetAllocationsSET spent\_amount = spent\_amount - 50000WHERE department = 'Research' AND id = 1;UPDATE BudgetAllocationsSET spent\_amount = spent\_amount + 50000WHERE department = 'Conservation' AND id = 2;IF @@ERROR\_COUNT > 0 THEN ROLLBACK; SELECT 'Transaction failed - rolled back';ELSE COMMIT; SELECT 'Transaction completed successfully';END IF;



Data Control Language (DCL)

User Privilege Management

-- Create UserCREATE USER 'forest\_manager'@'localhost' IDENTIFIED BY 'secure\_password';-- Grant Specific PrivilegesGRANT SELECT, INSERT, UPDATE ON

ForestDepartment.Staff TO 'forest\_manager'@'localhost';GRANT SELECT ON ForestDepartment.ForestZones TO 'forest\_manager'@'localhost';GRANT EXECUTE ON PROCEDURE AddPatrolRecord TO 'forest\_manager'@'localhost';-- Revoke PrivilegesREVOKE DELETE ON ForestDepartment.WildlifeSpecies FROM 'forest\_manager'@'localhost';

Control database access through granular permission assignment and revocation for different user roles.

Create User

1

Establish new database user with

secure credentials

Grant Privileges

2

Assign specific permissions for

tables and procedures

Revoke Access

3

Remove permissions when necessary for security





Trigger Implementation

Automate database responses to events with triggers that execute before or after INSERT, UPDATE, or DELETE operations.



1

Species Update Audit Trigger

Automatically logs population changes to audit table after species updates, capturing old and new values with timestamp and user information.

DELIMITER //CREATE TRIGGER after\_species\_updateAFTER UPDATE ON WildlifeSpeciesFOR EACH ROWBEGIN IF

OLD.population\_estimate !=

species\_id, old\_population, NEW.id, OLD.population\_estimate,

IF;END //DELIMITER ;

NEW.population\_estimate THEN

new\_population, changed\_by, NEW.population\_estimate,

INSERT INTO SpeciesAudit (

change\_date

USER(), NOW()

) VALUES (

); END

2

Automatic Timestamp Trigger

Updates timestamp field automatically before any staff record modification, ensuring accurate tracking of data changes.

CREATE TRIGGER before\_staff\_updateBEFORE UPDATE ON StaffFOR EACH ROWSET NEW.updated\_at = NOW();



##### Phase 5: Real-World Implementation

Deploying 100 comprehensive queries across real-world scenarios to create a

production-ready system for wildlife conservation and forest management.

100 Queries

Comprehensive operations covering DDL, DML, DQL, joins, subqueries, functions, views, procedures, transactions, and triggers.

Real Scenarios

End-to-end database operations simulating actual conservation challenges and management needs.

Production Ready

Performance-optimised system ready for deployment in active forest

conservation environments.



Wildlife Monitoring Dashboard

Comprehensive Species Tracking

SELECT wz.name, ws.species\_type, COUNT(DISTINCT ws.id),

SUM(ws.population\_estim ate), MAX(at.last\_seen)FROM ForestZones wzLEFT JOIN WildlifeSpecies ws ON wz.id = ws.zone\_idWHERE wz.status = 'Active'GROUP BY wz.nameORDER BY total\_population DESC;

Real-time wildlife monitoring across forest zones, tracking species populations, sighting data, and conservation status.

* Species count and population estimates by

zone

* Average population per species type
* Last sighting timestamps for active tracking
* Zone-wise wildlife distribution analysis



## Patrol Efficiency Analysis

50+

Average Area Covered

Square kilometres

patrolled per team

10%

Incident Rate

Threshold for excellent performance

6

Months Analysed

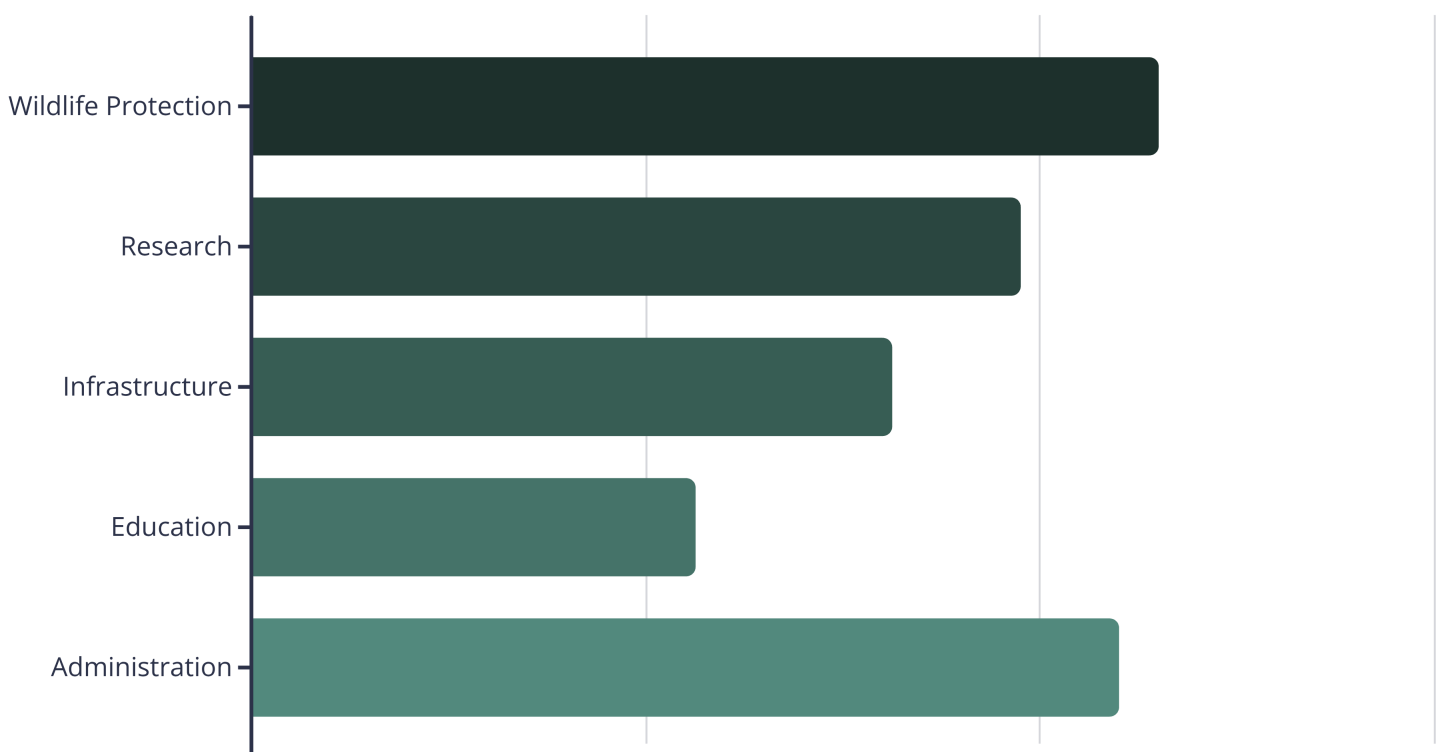
January to June 2025 patrol data



Performance rating system evaluates patrol teams based on area coverage and incident detection rates. Teams covering over 50 km² with incident rates below 10% receive 'Excellent' ratings, whilst those covering 30+ km² with rates under 20% achieve 'Good' status.



Budget Optimisation



###### Conservation Project Impact

Project Planning

1

2

3

4

Objectives defined, budget allocated, lead scientist assigned

Research Activities

Field studies conducted, species monitored, data collected

Impact Assessment

Duration tracked, species coverage analysed, outcomes evaluated

Status Reporting

Projects classified as completed, ongoing, or upcoming

Comprehensive analysis tracks project duration, research activities, species coverage, and budget utilisation. Projects are automatically categorised by status based on start and end dates relative to current date.

##### Automated Reporting System

The

GenerateMonthlyReport

Wildlife Population

Monthly population changes tracked with old vs new estimates and calculated differences.

Patrol Efficiency

Team performance metrics including patrols completed and average area coverage.

Budget Utilisation

Department-wise allocation, spending, and utilisation percentage calculations.

stored procedure consolidates critical

conservation metrics into comprehensive monthly reports, automatically calculating population changes, patrol efficiency, and budget utilisation for specified reporting periods.



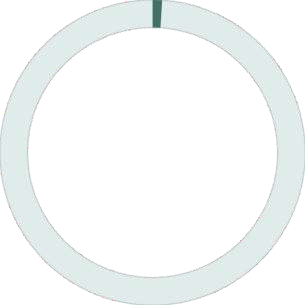
Disaster Management: Fire Risk Assessment



35°C



30%



1

|  |  |  |
| --- | --- | --- |
| High Risk Threshold | Humidity Threshold | Year Analysis |
| Temperature trigger for elevated fire danger | Low humidity increases fire susceptibility | Historical fire incident tracking period |
| High Risk | Medium Risk | Low Risk |
| Temperature >35°C and humidity <30% | Temperature 30-35°C and humidity 30-50% | All other conditions |

Real-time fire risk assessment combines current weather data with historical fire incidents to prioritise zones requiring enhanced monitoring and

rapid response capabilities.



## Eco-Tourism Impact Analysis

Visitor Tracking

Total visitors and purposes logged over six-month periods for active tourism spots.

Feedback Analysis

Average ratings calculated to assess visitor satisfaction and experience quality.

Revenue Generation

Total revenue computed from entry fees, supporting conservation funding initiatives.

Comprehensive tourism analysis evaluates visitor numbers, satisfaction ratings, and revenue generation for open eco-

tourism spots, enabling data-driven decisions on facility management and conservation funding.

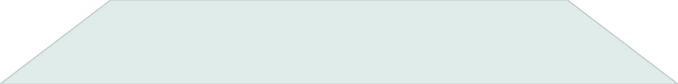
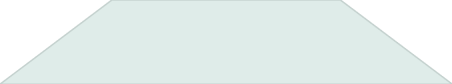


### System Architecture & Achievements

Frontend Layer



1



2

Application Layer

3

Database Layer

Phase 5 Complete

* 100+ real-world queries implemented
* Automated reporting system deployed
* Disaster management framework active
* Tourism impact analysis operational
* Production-ready database solution

Scalability Features

* Horizontal partitioning enabled
* Read replicas configured
* Caching mechanisms implemented
* Load balancing optimised
* 25 normalised tables with advanced indexing





## Environmental Impact



Biodiversity Conservation

* Real-time species monitoring
* Population trend analysis
* Habitat protection strategies
* Endangered species tracking

Climate Change Mitigation

* Carbon sequestration tracking
* Deforestation monitoring
* Afforestation programme management
* Climate resilience planning

Economic Benefits

Cost Savings



* Automated reporting reduces manpower
* Efficient resource allocation
* Preventive maintenance

scheduling

* Budget optimisation

Revenue Generation

* Eco-tourism management
* Sustainable resource utilisation
* Grant and funding tracking
* Carbon credit management



## Social Impact



Community Engagement

* Local employment opportunities
* Environmental education programmes
* Community-based conservation
* Tribal rights protection

Public Safety

* Disaster early warning systems
* Wildlife conflict management
* Fire prevention strategies
* Emergency response coordination



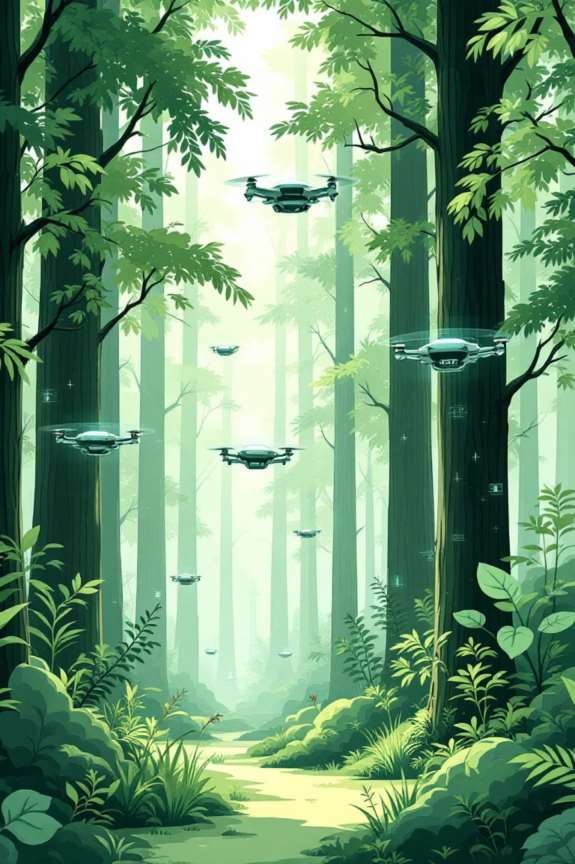
##### Key Learnings & Skills Developed

preencoded.pngTechnical Skills

* + Advanced SQL query writing
  + Database design & normalisation
  + Performance optimisation
  + Security implementation
  + Transaction management

preencoded.pngProfessional Skills

* + Project planning & execution
  + Documentation standards
  + Problem-solving approach
  + Real-world scenario analysis
  + Presentation & communication

Conclusion & Future Enhancements



Project Success

Comprehensive forest management solution with scalable architecture, real-world problem-solving approach, and professional implementation standards.



Future Enhancements

•

•

•

•

•

AI-powered predictive analytics

Mobile application integration IoT sensor integration

Real-time satellite data feeds

Blockchain for transparent funding

"Empowering forest conservation through technology – one query at a time!"

