Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)
School of Computing
B.Tech. – Computer Science and Engineering

VTR UGE2021- (CBCS)

Academic Year: 2025–2026
SUMMER SEMESTER - SS2526

Course Code : 10211CS207
Course Name: Database Management Systems
Slot No : S2L5

DBMS TASK - 12 REPORT

Title: Backing up and recovery in databases

Submitted by:

VTUNO	REGISTER NUMBER	STUDENT NAME
VTU27661	24UECS1159	C Venugopal reddy

Task 3: Online Examination Management System

6. Identify Entities

Entity	Attributes	Description
Student	student_id (PK), name, email, password, course	Stores student details
Exam	exam_id (PK), exam_name, date, duration, total_marks	Stores exam details
Question	question_id (PK), exam_id (FK), question_text, option_a, option_b, option_c, option_d, correct_option	Contains questions for each exam
Result	result_id (PK), student_id (FK), exam_id (FK), score, rank	Stores performance results

Relationships: A Student can take many Exams, an Exam contains multiple Questions, and a Result links a Student and an Exam.

7. Normalization

Normalization steps applied to reduce redundancy and ensure data integrity.
1NF: Each field is atomic and contains unique data.
2NF: Removed partial dependencies between attributes.
3NF: Removed transitive dependencies to keep only key-based attributes.
Normalized Tables: Student, Exam, Question, Result.

8. SQL Queries

a) Retrieve Student Scores:

```
SELECT s.student_id, s.name, e.exam_name, r.score
FROM Result r
JOIN Student s ON r.student_id = s.student_id
JOIN Exam e ON r.exam_id = e.exam_id;
```

b) Rank Students by Performance:

```
SELECT s.name, e.exam_name, r.score,
RANK() OVER (PARTITION BY e.exam_id ORDER BY r.score DESC) AS rank
FROM Result r
JOIN Student s ON r.student_id = s.student_id
JOIN Exam e ON r.exam_id = e.exam_id;
```

9. ACID Properties in Online Submissions

Property	Explanation
Atomicity	Each submission is saved completely or not at all.
Consistency	Database rules like score ≤ total_marks are enforced.
Isolation	Parallel submissions don’t affect each other.
Durability	Data remains safe even after a crash or restart.

10. CRUD Operations using MongoDB

Create:

```
db.results.insertOne({ student_id:101, exam_id:501, score:85, rank:2 });
```

Read:

```
db.results.find({ student_id:101 });
```

Update:

```
db.results.updateOne({ student_id:101, exam_id:501 }, { $set:{ score:90 }
});
```

Delete:

```
db.results.deleteOne({ student_id:101, exam_id:501 });
```

Summary: This system efficiently manages students, exams, questions, and results using normalized relational design, SQL ranking queries, ACID compliance, and MongoDB CRUD operations.