|  |  |
| --- | --- |
| **Course Code:21AI44** | **Date:** |
| **Sem: IV** | **Duration:** 90 Minutes |
| **CIE-III**  **Data Base Management Systems** | |

**Answer all Questions**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SL. No** | | **Questions** | **M** | **BT** | **CO** |
| 1 | a) | Apply five aggregate functions for the data set given in the relation and analyze the results   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Fname** | **Minit** | **Lname** | **SSN** | **Bdate** | **Sex** | **Salary** | **Super\_ssn** | **DNO** | | Killua | B | Zoldyck | 1234 | 1965-01-09 | M | 30000 | 3334 | 5 | | Mark | T | Lee | 3334 | 1955-11-09 | M | 40000 | 9876 | 5 | | Momo | J | Hirai | 9998 | 1968-01-19 | F | 25000 | 8886 | 4 | | Sunmi | S | Lee | 9876 | 1978-06-19 | F | 43000 | 8886 | 4 | | Sehun | K | Oh | 6668 | 1978-05-29 | M | 38000 | 3334 | NULL |  1. Sum (b) Minimum (c) Maximum (d) Average (e) count   Applications of these to the given data – 5\*2 = 10 | 10 | 2 | 1 |
| 2 | a) | Discuss the storage architecture of column oriented data bases?  Column-oriented databases are among the most popular types of non-relational databases   * Three key themes emerged   Data needs to be stored in a networked filesystem that can expand to multiple machines. Files themselves can be very large and be stored in multiple nodes, each running on a separate machine.  Data needs to be stored in a structure that provides more flexibility than the traditional normalized relational database structures. The storage scheme needs to allow for effective storage of huge amounts of sparse data sets. It needs to accommodate for changing schemas without the necessity of altering the underlying tables.   * Data needs to be processed in a way that computations on it can be performed in isolated subsets of the data and then combined to generate the desired output. This would imply computational efficiency if algorithms run on the same locations where the data resides. * It would also avoid large amounts of data transfer across the network for carrying out the computations on the humungous data set. | 06 | 2 | 1 |
|  | b) | Explain how column databases can be used as nested maps?   * Oftentimes, it’s easier to think of column databases as a set of nested maps. * Maps or hash maps, which are also referred to as associative arrays, are pairs of keys and their corresponding values. * Keys need to be unique to avoid collision and values can often be any array of bytes. * Some maps can hold only string keys and values but most column databases don’t have such a restriction. * Viewing the running example as a multidimensional nested map, you could create the first two levels of keys in JSON-like representation, like so: * The first-level key is the row-key that uniquely identifies a record in a column database. * The second-level key is the column-family identifier. * Three column-families — name, location, and preferences — were   defined earlier.   * Those three appear as second-level keys. * Going by the pattern, that the third-level key is the column identifier. * Each row may have a different set of columns within a column-family, so the keys at level three may vary between any two data points in the multidimensional map. * Adding the third level, the map is like so: | 04 | 2 | 1 |
| 3 | a) | Discuss the features of Apace Cassandra?   * **High Scalability** * Cassandra is highly scalable which facilitates you to add more hardware to attach more customers and more data as per requirement. * **Rigid Architecture** * Cassandra has not a single point of failure and it is continuously available for business-critical applications that cannot afford a failure. * **Fast Linear-scale Performance** * Cassandra is linearly scalable. It increases your throughput because it facilitates you to increase the number of nodes in the cluster. Therefore it maintains a quick response time. * **Fault tolerant** * Cassandra is fault tolerant. Suppose, there are 4 nodes in a cluster, here each node has a copy of same data. If one node is no longer serving then other three nodes can served as per request. * **Flexible Data Storage** * Cassandra supports all possible data formats like structured, semi-structured, and unstructured. It facilitates you to make changes to your data structures according to your need. * **Easy Data Distribution** * Data distribution in Cassandra is very easy because it provides the flexibility to distribute data where you need by replicating data across multiple data centers. * **Fast writes** * Cassandra was designed to run on cheap commodity hardware. It performs blazingly fast writes and can store hundreds of terabytes of data, without sacrificing the read efficiency. | 04 | 1 | 1 |
|  | b) | Discuss the following operations in Cassandra and also discuss the commands used in the operations?   1. Create the Keyspace AIML   Create keyspace KeyspaceName with replicaton={'class':strategy name,  'replication\_factor': No of replications on different nodes}   1. Alter the Keyspace AIML by increasing the replication factor to 4   Alter Keyspace KeyspaceName with replication={'class':'StrategyName',  'replication\_factor': no of replications on different nodes} with DURABLE\_WRITES=true/false   1. Create a table student   CREATE TABLE|COLUMN FAMILY<tablename> column-definition>’,’<column-definition>’) (WITH <option> AND <option>) | 06 | 1 | 2 |
| 4 | a) | Consider the following relational database schema  Student (Student-id,Sname,major,GPA)  Faculty (Faculty-id,fname,dept,designation,salary)  Course (Course-id,Cname,Faculty-id)  Enrol (Course-id,Student-id,grade)  Write the following queries in SQL:   1. List the names of all students enrolled for the course “IS6T1”.   SELECT S.Sname  FROM Student S  JOIN Enrol E ON S.Student-id = E.Student-id  JOIN Course C ON E.Course-id = C.Course-id  WHERE C.Cname = 'IS6T1';   1. List the names of all students enrolled for the course “IS6T1 and have received “A” grade.   SELECT S.Sname  FROM Student S  JOIN Enrol E ON S.Student-id = E.Student-id  JOIN Course C ON E.Course-id = C.Course-id  WHERE C.Cname = 'IS6T1' AND E.grade = 'A';   1. List all the departments having an average salary of above Rs. 10,000.   SELECT F.dept  FROM Faculty F  GROUP BY F.dept  HAVING AVG(F.salary) > 1000;   1. Give a 20% raise to salary of all faculty.   UPDATE Faculty  SET salary = salary \* 1.20;   1. Find the maximum, minimum and average salary of all the faculty.   SELECT MAX(salary) AS max\_salary, MIN(salary) AS min\_salary, AVG(salary) AS avg\_salary  FROM Faculty; | 10 | 4 | 3 |
| 5 |  | Analyse all possible constraint violations that can occur during insert operation with suitable examples  **Primary Key Violation**:   1. A primary key constraint ensures that a column or set of columns uniquely identify each row in a table. 2. Violation: Trying to insert a row with a duplicate primary key value.   **Unique Constraint Violation**:   * A unique constraint ensures that values in a specific column (or set of columns) are unique across the table. * Violation: Trying to insert a row with a value that already exists in the unique column.   **Foreign Key Violation**:   * A foreign key constraint ensures referential integrity between two tables. * Violation: Trying to insert a value into a foreign key column that does not exist in the referenced table's primary key.   **Check Constraint Violation**:   * A check constraint defines a condition that must be satisfied for each row in a table. * Violation: Trying to insert a row that does not satisfy the specified condition.   **Not Null Constraint Violation**:   * A not null constraint ensures that a column cannot have null values. * Violation: Trying to insert a row with a null value in a not null column. | 10 | 2 | 1 |

|  |  |
| --- | --- |
| **Course Outcome** | |
| **CO1** | Understand and Apply Database Management Systems concepts to solve the given problem |
| **CO2** | Design solutions with societal and environmental concerns using modern tools to solve problems in Database Design domain |
| **CO3** | Analyze and develop Database Applications using SQL and NoSQL features by engaging in lifelong learning for emerging technology |
| **CO4** | Exhibit effective communication and engage in continuing professional development through experiential learning |
| **CO5** | Demonstrate skills like investigation, effective communication, working in team/Individual practices by implementing Database Design concepts and applications |

**M-Marks, BT-Blooms Taxonomy Levels, CO-Course Outcomes**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Marks Distribution** | **Particulars** | **CO1** | **CO2** | **CO3** | **CO4** | **L1** | **L2** | **L3** | **L4** | **L5** | **L6** |
| **Max Marks** | 34 | 06 | 10 | -- | -- | 10 | 30 | -- | 10 | -- |