INSTITUTE OF TECHNOLOGY TRALEE

WINTER EXAMINATION AY 2017/ 2018

Advanced Database Programming

CRN 48064

External Examiner: Sean McHugh
Internal Examiner: Mr. Peter Given

Duration: 2 Hours

Instructions to Candidates:

i) Answer any **three** questions.

ii) All questions carry equal marks. Submit all your rough-work, marks may be lost otherwise.

Question 1:

i) A CouchDB database contains JSON documents that contain a field for "*band name*" along with a field for "*albums*" which contains an array of embedded JSON objects which contain information about the album's "*name*" and "*year released*". Explain how you would write a query in CouchDB to find out how many albums a particular band has released and explain how the query works. **(13 marks)**

```
function(doc) {
 (doc.albums || []).forEach(function(album){
        emit(doc.name, 1);
});
}
function(key, values, rereduce) { return sum(values);
}
```

ii) In comparison to a relational database such as PostgreSQL, discuss the weaknesses of the database design in part i). **(10 marks)**

iii)

iv) iii) Write a note on a) JSON and b) conflict resolution in CouchDB. **(10 marks)**

Question 2:

i) Compare and contrast MongoDB with CouchDB, using examples where appropriate.

(12 marks)

Which database is right for your business?		
	CouchDB	**MongoDB**
Speed	If read speed is critical to your database, MongoDB is faster than CouchDB.	MongoDB provides faster read speeds.
Mobile support	CouchDB can be run on Apple iOS and Android devices, offering support for mobile devices.	No mobile support provided.
Replication	Offers master-master and master-slave replication.	Offers master-slave replication.
Size	While your database can grow with CouchDB, MongoDB is better suited for rapid growth when the structure is not clearly defined from the beginning.	If you have a rapidly growing database, MongoDB is the better choice.
Syntax	Queries use map-reduce functions. While it may be an elegant solution, it can be more difficult for people with traditional SQL experience to learn.	For users with SQL knowledge, MongoDB is easier to learn as it is closer in syntax.
Analysis	If you need a database that runs on mobile, needs master-master replication, or single server durability, then CouchDB is a great choice.	If you need maximum throughput, or have a rapidly growing database, MongoDB is the way to go.

	Couchdb	Mongodb
Data Model	Document-Oriented (JSON)	Document-Oriented (BSON)
Interface	HTTP/REST	Custom protocol over TCP/IP
Object Storage	Database contains Documents	Database contains Collections Collections contains Documents
Query Method	Map/Reduce (javascript + others) creating Views + Range queries	Map/Reduce (javascript) creating Collections + Object-Based query language
Replication	Master-Master with custom conflict resolution functions	Master-Slave
Concurrency	MVCC (Multi Version Concurrency Control)	Update in-place
Written In	Erlang	C++

ii) "*MongoDB is an easy transition for developers used to working with the relational model*". Discuss. **(12 marks)**

1. **** Data Models: **** A NoSQL database lets you build an application without having to define the schema first unlike relational databases which make you define your schema before you can add any data to the system. No predefined schema makes NoSQL databases much easier to update as your data and requirements change.

2. **** Data Structure: **** NoSQL databases are designed to handle unstructured data (e.g., texts, social media posts, video, email) which makes up much of the data that exists today. Relational databases were built in an era where data was fairly structured and clearly defined by their relationships.

3. **** Scaling: **** It's much cheaper to scale a NoSQL database than a relational database because you can add capacity by scaling out over cheap, commodity servers. Relational databases, on the other hand, require a single server to host your entire database. To scale, you need to buy a bigger, more expensive server

4. **** Development Model: **** NoSQL databases are open source whereas relational databases typically are closed source with licensing fees baked into the use of their software. With NoSQL, you can get started on a project without any heavy investments in software fees upfront.

SQL Database	NoSQL Database (MongoDB)
Relational database	Non-relational database
Supports SQL query language	Supports JSON query language
Table based	Collection based and key-value pair
Row based	Document based
Column based	Field based
Support foreign key	No support for foreign key
Support for triggers	No Support for triggers
Contains schema which is predefined	Contains dynamic schema
Not fit for hierarchical data storage	Best fit for hierarchical data storage
Vertically scalable - increasing RAM	Horizontally scalable - add more servers
Emphasizes on ACID properties (Atomicity, Consistency, Isolation and Durability)	Emphasizes on CAP theorem (Consistency, Availability and Partition tolerance)

iii) Discuss how and why Mongo uses Replica sets, giving examples where appropriate.

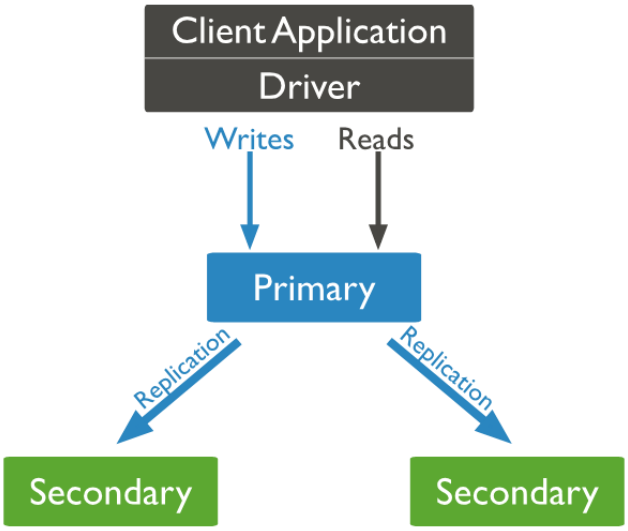
(9 marks)

- **Replication** is the process of creating and managing duplicate versions of a database, which provides **redundancy** and increases **availability** of data.

- It allows you to **keep copies of your data** in redundancy, and these can then be used for vital purposes, such as ensuring a backup that can maintain your system's availability, or recovering your data if your primary instance ever fails.

- MongoDB initiates this replication process using replica set. A replica set contains several **data bearing nodes** and o**ptionally one arbiter node.** Of the data bearing nodes, one and only one member is deemed the primary node, while the other nodes are deemed secondary nodes.

- The primary node receives all **write operations.** A replica set can have only one primary capable of **confirming writes concern**. The primary records all changes to its data sets in its operation log.



Advantages of Replication:-

1. Increased **reliability** and **availability**
2. Queries requesting replicated copies of data are always **faster**
3. **Less Communication** Overhead

For example,

- An application might normally access a local database rather than a remote server to minimize network traffic and achieve maximum performance.
  The application can continue to function if the local server experiences a failure, but other servers with replicated data remain accessible.

Question 3:

i) Discuss the strengths and weaknesses of Neo4J and describe a use case for an application which would be suited to using a Neo4J database. **(13 marks)**

Strengths

- Graph Databases are good for unstructured data in many more ways than document databases.
- Neo4j is typeless and schemaless but puts no constraints on how data is related.
- Neo4j is fast unlike joint operations in document databases such as map reduce queries , graph traversals are in constant time.

Weaknesses

- Relationships(Edges) in Neo4j cannot direct a vertex back at itself.
- Although HA is good at replication it can only replication a full graph to another server.
- If you are looking for a business friendly open source db Neo4j is probably not the best option for you.

ii) Appendix 1 shows the Cypher code used to create a Neo4j database. Draw a graph representing the resulting database and show queries that will retrieve a) the names of all the people Alice is friends with b) friends of friends of Alice **(10 marks)**

iii) Discuss the CAP theorem and explain where Neo4J High Availability sits in relation to this theory. **(10 marks)**

Question 4

i) Discuss the strengths and weaknesses of HBase. **(12 marks)**

ii) Compare and contrast a HBase table with a table in a relational database such as PostgreSQL. **(12 marks)**

iii) "*With respect to CAP, HBase is decidedly CP*". Discuss. **(9 marks)**

Appendix 1: Cypher Code to create a Neo4j Database

```
CREATE (p:Publication {name: "Wine Expert Monthly"})

MATCH (p:Publication {name: "Wine Expert Monthly"}),
(w:Wine {name: "Prancing Wolf", vintage: 2015}) CREATE
(p)-[r:reported_on]->(w)

MATCH (p:Publication {name: "Wine Expert Monthly"}),
(w:Wine {name: "Prancing Wolf"})
CREATE (p)-[r:reported_on {rating: 97}]->(w)

CREATE (g:GrapeType {name: "Riesling"})

MATCH (w:Wine {name: "Prancing Wolf"}),(g:GrapeType {name: "Riesling"})
CREATE (w)-[r:grape_type]->(g)

CREATE (wr:Winery {name: "Prancing Wolf Winery"})

MATCH (w:Wine {name: "Prancing Wolf"}),(wr:Winery {name: "Prancing Wolf Winery"})
CREATE (wr)-[r:produced]->(w)

CREATE (w:Wine {name:"Prancing Wolf", style: "Kabinett", vintage: 2002});

CREATE (w:Wine {name: "Prancing Wolf", style: "Spätlese", vintage: 2010});

MATCH (wr:Winery {name: "Prancing Wolf"}),(w:Wine {name: "Prancing Wolf"}) CREATE
(wr)-[r:produced]->(w)

MATCH (w:Wine),(g:GrapeType {name: "Riesling"})
CREATE (w)-[r:grape_type]->(g)

CREATE (p:Person {name: "Alice"})

MATCH (p:Person {name: "Alice"}),
(w:Wine {name: "Prancing Wolf", style: "ice wine"})
CREATE (p)-[r:likes]->(w)

CREATE (p: Person {name: "Tom"})

MATCH (p:Person {name: "Tom"}),(w:Wine {name: "Prancing Wolf", style: "ice wine"})
CREATE (p)-[r:likes]->(w)

MATCH (p:Person {name: "Tom"}),
(pub:Publication {name: "Wine Expert Monthly"}) CREATE
(p)-[r:trusts]->(pub)

CREATE (p:Person {name: "Patty"})

MATCH (p1:Person {name: "Patty"}),
(p2:Person {name: "Tom"})
```

```
CREATE (p1)-[r:friends]->(p2)

MATCH (p1:Person {name: "Patty"}),
(p2:Person {name: "Alice"})
CREATE (p1)-[r:friends]->(p2)

CREATE (p1:Person {name: "Ahmed"}), (p2:Person {name: "Kofi"});


MATCH (p1:Person {name: "Ahmed"}),(p2:Person {name: "Alice"})
CREATE (p1)-[r:friends]->(p2);

MATCH (p1:Person {name: "Kofi"}),(p2:Person {name: "Tom"});
CREATE (p1)-[r:friends]->(p2);
```