**1. HBase Denormalization with Consistency**

Emails received from different users for an particular account are stored in an email table.

The attributes of the email table are:

userId

messageId

timestamp

email-message

attachmentId

Design an HBase schema for the below requirements such that it provides excellent random read performance when:

Retrieving all emails for a given user ID and date

Retrieving all attachments for a given email for a given user ID and date

Please include sample queries showing both of these types of access, and the row key design employed for each requirement.

**SOLUTION:**

**Table and rowkey design**

table name : emails

row key : <userId><dateId><messageId>

column family : data

columns : timestamp, message, attachmentId

**Using Phoenix:**

create table emails (userId int not null, dateId int not null, messageId int not null, timestamp datetime, message text, attachemntId int, CONSTRAINT pk PRIMARY KEY (userId, dateId, messageId));

1. SELECT messageId, message FROM emails WHERE userId = <> AND dateId = <> ;

Explanation: This goes for a range scan with prefix filter because a part of the row key is used in the predicates.

1. SELECT attachmentId FROM emails WHERE userId = <> AND dateId = <> AND messageId = <> ;

Explanation: This goes for a point get as the entire rowkey is used in the predicate

**Using hbase shell**:

1. scan 'emails' , {COLUMN => ['data:messageId','data:message'] }, { FILTER => " PrefixFilter('<userId><dateId>') "}

2. get 'emails' , '<userId><dateId><messageId>', {COLUMN => 'data:attachmentId'}

The queries are self explanatory.

**2. Slowly-Changing Dimensions**

Users post questions on the a legal forum, and the forum's attorneys answer those questions based on their practice area. All the questions and answers posted will be reviewed and assigned a status code.

Here are different status codes for reference:

0: Decision Pending

1: Question or Answer Accepted

2: Question or Answer Declined

A question or answer will initially have status code "0" and change to either "1" or "2". It is also possible that the status for a question or answer could shift between "1" and "2" at any point of time.

We want a report that would measure each attorney's efficiency in the following ways:

Number of Decision Pending/Approved questions answered by day.

Count of Decision Pending/Approved answers and count of Declined answers by day.

Count of approved questions with approved answers answered in a month.

Please outline the database objects you would create to measure this. You can draw a diagram, create DDL, or any other method you would like to communicate what you build. Please include SQL statements that we could use to answer the above questions.

**SOLUTION:**

To answer the above queries, SCD is not necessary. These can be answered without storing the change history. Following is the table design and the SQL queries to answer respective query.

CREATE TABLE forumn\_questions (

questionid int NOT NULL AUTO\_INCREMENT,

Userid int NOT NULL,

Content text NULL,

Status int NOT NULL,

Created\_date datetime NOT NULL,

Updated\_date datetime NOT NULL,

PRIMARY KEY (questionid),

KEY (userid)

);

CREATE TABLE forumn\_answers (

answerid int NOT NULL AUTO\_INCREMENT,

Questionid int NOT NULL,

Userid int NOT NULL,

Content text NULL,

Status int NOT NULL,

Created\_date datetime NOT NULL,

Updated\_date datetime NOT NULL,

PRIMARY KEY (answerid),

KEY (questionid),

KEY (userid),

FOREIGN KEY (questionid) REFERENCES forumn\_questions (questionid)

);

1. **Number of Decision Pending/Approved questions answered by day.**

SELECT

A.userid, date(A.created\_date) as dt, count(distinct A.questionid) as questions

FROM forumn\_answers A

JOIN forumn\_questions B ON A.questionid = B.question\_id

WHERE B.status in (0, 1)

GROUP BY A.userid, date(A.created\_date)

ORDER BY userid, created\_date;

**b) Count of Decision Pending/Approved answers and count of Declined answers by day.**

SELECT

userid, date(created\_date) as dt, status, sum(answerid) as count

FROM forumn\_answers

GROUP BY userid, date(created\_date), status

ORDER BY userid, created\_date, status;

**c) Count of approved questions with approved answers answered in a month.**

SELECT

A.userid, year(A.created\_date) as yr, month(A.created\_date) as mth, count(distinct A.questionid) as questions, count(A.answerid) as answers

FROM forumn\_answers A

JOIN forumn\_questions B ON A.questionid=B.questionid

WHERE A.status = 1 and B.status=1

GROUP BY A.userid, year(A.created\_date), month(A.created\_date)

ORDER BY user\_id, created\_date ;

**3. Tableau**

Tableau is our reporting layer and we have automated a lot of our standard reporting on Tableau Server. Please answer the following questions and focus your answers on Tableau.

You are checking on an automated report on Tableau server set to run each business day. When you open the report, the filters are still present, but there are no results (no data, no viz). There are also no errors. What are some of the possible explanations for why there would not be data present in the report? Describe how you would check for each possible explanation you provide.

We produce Tableau reports that connect to multiple servers. Assume that one of the servers went down, it’s unclear when it will be back online, and that this server is the most widely used data source. What would you do immediately to minimize the impact on business customers?

**SOLUTION:**

I haven’t used tableau before, but possible explanations can be

1. No result set for the filter matching criteria. This can be due to data not being populated in the underlying reporting store. Check for the job logs for the run that populates data, debug or trace the data lineage with record counts at each stage for the selected criteria.
2. This may be tableau specific. .so no idea as i haven’t used it.

**4. Scala: A Simple Calculator**

In Scala, write a method called calculator that accepts three string parameters:

def calculator(operand1: String, operator: String, operand2: String): Unit

and demonstrate that it behaves as follows

Converts the operands to Ints;

Performs the desired mathematical operator (+, -, \*, or /) on the two operands

Prints the result, or a generic error message

**SOLUTION**

def calculator(operand1: String, operator: String, operand2: String): Unit = {

try {

operator match {

case "+" => println(operand1.toInt + operand2.toInt)

case "-" => println(operand2.toInt - operand2.toInt)

case "\*" => println(operand2.toInt \* operand2.toInt)

case "/" => println(operand2.toInt / operand2.toInt)

case err => println("Generic Error Message")

}

}

catch {

println("Generic Error Message")

}

}

**5. Query Optimization**

Consider the following SQL query:

Select a. \* , b. \* , c. \* , d. \*

from table\_a a left join (select \*, case when column\_1b\_sub > 0 then 1 else 0 end as column\_1b\_sub\_value from table\_b) b on a.column\_1a = b.column\_1b and a.column\_2a = b.column\_2b

left join (select \* from table\_d) d on a.column\_1a = b.column\_1b and a.column\_2a = b.column\_2b

Identify any syntactic problems with this query. With those problems fixed, how could you modify this query to make it more efficient?

**SOLUTION:**

Below observations are based on the ANSI SQL standards and the implicit assumption that the mentioned columns exists in the corresponding table definitions.

1. table alias c used in select, does not exists in the query

2. The last left join is syntactically correct, but illogical, will result in cartesian join, since the join condition is different from joining table.

**Optimized:**

1. Used actual table since entire table was present in sub-query.

2. Remove the join condition for last left join, because it was redundant. Since it was a cartesian join and the join condition was already executed on the previous line, it would produce the same result set.

SELECT a. \* , b. \* , c. \* , d. \*

FROM table\_a a LEFT JOIN

(SELECT \*, CASE WHEN column\_1b\_sub > 0 THEN 1 ELSE 0 END as column\_1b\_sub\_value FROM table\_b) b ON a.column\_1a = b.column\_1b AND a.column\_2a = b.column\_2b

LEFT JOIN table\_d d