Weekly exercises IN3020&4020 (2026)
Week 12: Security and NoSQL Intro

1 Security

1. Discuss the main issues that we need to deal with in database security as well as the main threats.

   Solution:

   Issues are legal, ethical, and policy; system-related; and security categorisation and organisation.

   Threats are loss of integrity, loss of availability, loss of confidentiality

2. Introduce discretionary and mandatory security mechanisms for multi-user DBMSs.

   Solution:

   Discretionary security mechanisms are used to grant privileges to users to access certain pieces of data; commonly used in practice.

   Mandatory security mechanisms classify data pieces and users into various security classes, and give access according to a corresponding security policy; used (and extended) in professional DBMSs.

3. Discuss main control measures for data security.

   Solution:

   Access control (in narrow sense) is handled by user accounts (with passwords) and data labelling to ensure the security mechanism.

   Inference control ensures that information about individuals cannot be derived key in statistical databases

   Flow control prevents information from flowing from authorised to unauthorised users disallow covert channels.

   Data encryption is used to protect sensitive data during transmission symmetric/asymmetric encryption, digital signatures and certificates

4. What is the trade-off between security and precision?

   Solution: Ideally we would like to have a system with perfect security and maximum precision, this is in contradictory. In practice we need to find the best trad-off between the two. E.g., we want to keep the data sufficiently secure, but we would also like to make it available for use.

5. What is the difference between security and privacy?

   Solution: Security handles issues related to access to resources, while privacy goes beyond, it handles issues related to appropriate use of information (i.e., the ability of individuals to control the terms under which their personal information is acquired and used).

6. Outline the main details of discretionary-based access control (DAC) in SQL.

   Solution: It is part of the SQL standard, which is based on granting and revoking privileges to users to access data (by means of commands `GRANT` and `REVOKE`). It has two levels of privileges: the account level and relation level. The account level specifies the privileges for defining, modifying and accessing the whole database, while relation level does nearly the same for specific tables and views. A special optional modifier allows to propagate relation-level privileges to other users; revoking is automatically cascading. Advantages of DAC is that it is part of the standard and quite simple; the disadvantages is that it can be challenging to maintain with large systems with many users.

7. (*) Consider a relation

   $$\textit{Employees}[\textit{Id, Name, Address, Age, Position, Salary}].$$

   Write SQL commands with DAC mechanism that give read privileges to *UserA* to the address of the employee with name 'John' as written in the relation (and to nothing else).

   Solution: For such fine-grained privileges we have to use views:

   ```
   CREATE VIEW ViewForUserA AS
          (SELECT Address FROM Employees WHERE Name='John')
   ```

   and give access to the information in the view only:

   ```
   GRANT SELECT ON ViewForUserA
   ```

8. Outline the main details of mandatory-based access control (MAC) in SQL, including Bell-LaPadula rules.

Solution: MAC is used in more professional systems, such as ORACLE. Both subjects (i.e., users) and objects (i.e., data elements) are classified into security classes, which are organised as a (partial) order. Two rules (Bell-LaPadula) are enforced:

- simple security says that no one can read data with higher security than his/her clearance
- star property says that no one ca write data with lower security than his/her clearance

In several SQL engines MAC is implemented with 4 security classes ($TS \geq S \geq C \geq U$) with objects being both attribute values in tuples and tuples themselves. The advantages are that it is easier to control access in large organisations and ensures flow control. The disadvantages are that it may be an overkill in smaller systems and it is not part of the SQL standard.

9. What do we mean by inference control? Give one example where this is important.

   Solution: Inference control prevents users to infer classified information based on available information. One example is using a statistical database where the user does not have the rights to to access individual data items. They could, however infer such data if they can specify a query that returns the average of one person. Another example is if the user can retrieve some information from a database and then by combining this information with publicly available data recreate sensitive information.

10. What is flow control?

    Solution: Regulation of the flow of information among accessible objects. It ensures that classified information does not flow into less classified objects

2 NoSQL DBMS

1. Name the main motivation points and challenges for NoSQL databases. What do we usually sacrifice in NoSQL to address the challenges in comparison to Relational Databases?

   Solution: The main motivation points are a need for creating and managing a large volume of semi-structured data distributed across several unreliable nodes, and the challenge is to ensure high performance, data availability and scalability, and sharding and partitioning. The

main sacrifices are loss of strong consistency (i.e., every read returns the most recent write) and weaker query languages (usually, only CRUD).

2. What are the V-characteristics of Big Data?

   Solution: The original three are Volume, Velocity, and Variety; however, there are more added later, which include Veracity, Value, and Variability.

3. What is the CAP theorem, why is this relevant for NoSQL databases?

   Solution: It is relevant to NoSQL because we have a distributed system with replicated data. CAP theorem (just a principle, no proof) states that If we have a distributed system with replicated data, we cannot at the same time ensure all three of the following properties:

   (a) Consistency (every read either get the most resent write)

   (b) Availability (every read and write receives a response)

   (c) Partition tolerance (works after a split of data to several disconnected parts)

4. Give the main categories of NoSQL approaches, describe their main ideas and system examples.

   Solution: A document-based system is a collection of similar documents. A document can be complex, self-describing, and have different data elements (e.g., in XML or JSON formats). One example is MongoDB which uses BSON (a variant of JSON) as a format.

   A (pure) key-value store keeps only pairs of keys (ids) and values (strings, etc.) It focuses on high performance, availability, and scalability and hence does not have query language, concentrating on search by key. Examples are DynamoDB and Voldemort, which use key-value ideas, but also a little more structure on data.

   A column-based system is similar to key-value store, but with more complex keys. In particular, it may be seen that table columns (i.e., attributes) are grouped into column families, which are stored in their own files. For example, Apache HBase is such a system.

   Finally, graph-based systems rely on graph representation of data. Their examples are RDF (with query language SPARQL) and Neo4j property graphs (with query language Cypher).

3 Try MongoDB (optional)

Setup:

1. To install MongoDB on your computer, follow the instructions for you system here:
   `https://www.mongodb.com/docs/manual/tutorial/`.
2. Copy the file municipalities.json from the web page of the course (same folder where this pdf is)
3. Load data into the database with the command
   `$ mongoload --db counties --collection places --file municipalities.json`
4. After installing and starting the service, you should be able to connect to the database with:
   `$ mongosh`
5. There is a nice overview of operations here: `https://www.mongodb.com/developer/quickstart/cheat-sheet/`.

Try out:

1. Start the shell
2. Show which databases you have by means of operation

   ```
   show dbs
   ```

3. Show which collections you have in the database by means of operation

   ```
   show collections
   ```

4. List all data by means of operation

   ```
   db.places.find()
   ```

5. List data about Akershus by means of operation

   ```
   db.places.find({"county" : "Akershus"})
   ```

6. Remove the entry “Oslo” by means of operation

   ```
   db.places.remove({"county" : "Oslo"})
   ```

7. Insert the entry “Oslo” by means of operation

   ```
   db.places.insert({"county" : "Oslo", "municipalities" : ["Oslo"]})
   ```

8. Find the county with the municipality “Hamar” (do a projection) by means of operation

   ```
   db.places.find({"municipalities": "Hamar"}, {"_id:0", "county":1})
   ```