# Database Administration (CST4714) – 15-Week Syllabus (Open Resources)

## Course Description

This course provides a comprehensive introduction to database administration, covering both relational and NoSQL database systems. Students will learn fundamental tasks and issues in database administration, including planning, installing, configuring, monitoring, tuning, securing, backing up, and troubleshooting databases. Emphasis is placed on hands-on administration of an SQL RDBMS (using PostgreSQL) in the first half, and a cloud-based NoSQL DBMS (MongoDB Atlas) in the second half. By working with these open platforms, students will gain practical experience in managing users and roles, optimizing queries, implementing backup/recovery strategies, and ensuring database security and performance[[1]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=PostgreSQL%20,a%20complete%20treatment%2C%20we%20recommend)[[2]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Database%20administrators%20are%20becoming%20more,got%20the%20answers%20you%20need). The course includes laboratory exercises and a term project to apply these skills in real-world scenarios.

## Course Objectives

Upon successful completion of this course, students will be able to: - **Roles & Responsibilities:** Explain the role of a database administrator and the key responsibilities involved in managing enterprise databases (e.g. design, deployment, maintenance, security)[[3]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Key%20responsibilities%20of%20a%20database,administrator)[[4]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Maintaining%20and%20troubleshooting%20databases).  
- **Database Installation & Configuration:** Install and configure a relational database server (PostgreSQL) and a cloud NoSQL database service (MongoDB Atlas), creating and initializing operational databases.  
- **Schema & Storage Management:** Manage physical and logical structures of databases, including schemas, tables, indexes, and storage (tablespaces, files) in an SQL DBMS, and effectively model data in a NoSQL document database[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design)[[6]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Embedded%20Data%20Model).  
- **User Management & Security:** Administer database security by creating users/roles, granting privileges, and implementing access controls on both SQL and MongoDB platforms[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning)[[8]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Considerations%20while%20designing%20Schema%20in,MongoDB).  
- **Performance Monitoring & Tuning:** Monitor database performance (memory, I/O, query execution) and identify tuning opportunities. Use tools like **EXPLAIN** in PostgreSQL to optimize query processing and adjust configuration for performance improvement[[9]](https://thoughtbot.com/blog/reading-an-explain-analyze-query-plan#:~:text=The%20most%20powerful%20tool%20at,SQL%20queries%20is%20EXPLAIN%20ANALYZE).  
- **Backup & Recovery:** Plan and implement basic backup and recovery procedures for SQL and NoSQL databases (e.g. using pg\_dump/pg\_restore in PostgreSQL and MongoDB’s mongodump/mongorestore utilities)[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server).  
- **NoSQL Specific Skills:** Deploy and manage a MongoDB Atlas cluster, design effective document schemas (embedded vs. referenced) for a given use case[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design), perform CRUD operations on MongoDB, create indexes for performance[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index), and configure replication/sharding for high availability and scalability.  
- **Database Monitoring & Maintenance:** Utilize administrative tools (like **pgAdmin**, **psql** for PostgreSQL, and **MongoDB Compass** or Atlas UI for MongoDB) to monitor database health, perform routine maintenance (vacuuming, index maintenance, etc.), and troubleshoot common issues.  
- **Project Execution:** Synthesize the above skills in a project or case study by planning, implementing, and documenting a database solution (including both relational and NoSQL components) to meet specified requirements.

## Prerequisites

Students are expected to have completed an introductory database course (equivalent to CST3604) with a grade of C or higher. This includes basic knowledge of relational database design, SQL querying, and normalization. Familiarity with fundamental operating system commands and networking is also helpful since we will install software and use cloud services.

## Required Texts and Materials (Open Access)

All course resources are free and openly available:

* **Database Design (2nd Edition)** – Adrienne Watt (BCcampus OpenEd). An open textbook covering database design concepts, data modeling, normalization, and SQL basics[[12]](https://opentextbc.ca/dbdesign01/#:~:text=Book%20Description)[[13]](https://opentextbc.ca/dbdesign01/#:~:text=5,Database%20development%20process). *Use this for review of database modeling and design in Week 1.*
* **PostgreSQL Wikibook** – *PostgreSQL* on Wikibooks (open-content book). Focuses on PostgreSQL architecture, installation, maintenance, and basic optimization tasks, aimed at novice DBAs[[1]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=PostgreSQL%20,a%20complete%20treatment%2C%20we%20recommend). *Primary reference for Weeks 2–7 topics (PostgreSQL administration).* Key sections include PostgreSQL **Architecture**[[14]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=PostgreSQL%20implements%20a%20client,at%20the%20server%20site), **Setup/Configuration**, **Roles and Privileges**[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning), **Backup & Recovery**[[15]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,Backup%20%26%20Recovery), etc.
* **PostgreSQL Official Documentation** (latest version, free online) – for in-depth reference on specific commands and tools (e.g. EXPLAIN, pg\_dump, configuration settings)[[16]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,which%20summarizes%20some%20special%20topics).
* **The Little MongoDB Book** – Karl Seguin. A concise introduction to MongoDB (document-oriented NoSQL) – *freely available under CC BY-NC license*. Covers MongoDB basics: CRUD operations, indexing, replication, sharding, etc. *(Use as a supplemental conceptual guide in Weeks 8–14.)*
* **MongoDB Online Manual** – MongoDB Inc. (free web documentation). Covers **MongoDB Atlas** setup and usage, and MongoDB server features. We will use specific sections on **CRUD Operations**[[17]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=MongoDB%20is%20a%20persistent%20document,fundamental%20types%20of%20data%20operations), **Data Modeling** (embedded vs. normalized documents)[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design), **Indexing**[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index), **Replication & Sharding**, **Backup/Restore**, and **Monitoring**.
* **Tutorialspoint MongoDB Tutorial** (free online resource) – Provides step-by-step guides for MongoDB tasks. Useful chapters include *Data Modeling*[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design), *Insert/Query/Update/Delete Documents*, *Indexing*[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index), *Replication* and *Sharding*, *Backup* (mongodump)[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server). These will be referenced for hands-on labs in Weeks 8–14.
* **Additional Readings:** Instructor may provide links to articles or blog posts for certain advanced topics (e.g. reading PostgreSQL **EXPLAIN** plans[[9]](https://thoughtbot.com/blog/reading-an-explain-analyze-query-plan#:~:text=The%20most%20powerful%20tool%20at,SQL%20queries%20is%20EXPLAIN%20ANALYZE), or a case study on MongoDB schema design). All provided materials will be openly accessible.

**Tools and Software:**  
- **PostgreSQL 15** (or latest stable) – open-source relational database system. Students should install PostgreSQL on their own computer or use a provided server.  
- **pgAdmin 4** – open-source graphical administration tool for PostgreSQL[[15]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,Backup%20%26%20Recovery). This will be used in labs for managing the PostgreSQL database (creating roles, running queries, viewing server status). **psql** (PostgreSQL’s command-line shell) will also be used for certain tasks.  
- **MongoDB Atlas** – cloud-based MongoDB service (free-tier cluster). We will use Atlas to create a MongoDB cluster starting Week 8. **MongoDB Compass** (free GUI for MongoDB) is recommended for visual exploration of data and query building. Alternatively, students can use the Atlas web interface or the **mongosh** shell for MongoDB operations.  
- **MongoDB CLI Tools:** *mongosh*, *mongodump*, *mongorestore* – used in labs for interacting with Atlas and performing backups/restores.  
- **Other:** A text editor or IDE of your choice for writing SQL scripts or shell commands, and a web browser to access online resources and the Atlas dashboard.

All software used is free. Detailed setup instructions will be provided in labs.

## Grading and Evaluation

* **Quizzes:** 20% – Short quizzes (4×5% each) to assess understanding of recent topics. (Tentatively in Weeks 3, 5, 10, 13.)
* **Midterm Exam:** 20% – Comprehensive exam in Week 7 covering Weeks 1–6 (PostgreSQL portion).
* **Final Exam:** 30% – Exam in Week 15 covering Weeks 8–14 (MongoDB portion, plus overarching concepts).
* **Case Study/Project:** 25% – A term project (in teams or individual) applying both PostgreSQL and MongoDB administration in a realistic scenario. This includes a written report and an in-class presentation/demonstration (due Week 14).
* **Homework/Lab Assignments:** 5% – Periodic homework or lab reports. These may include short write-ups of lab results or small research tasks.

**Grade Scale:** A = 93–100, A– = 90–92.9, B+ = 87–89.9, B = 83–86.9, B– = 80–82.9, C+ = 77–79.9, C = 70–76.9, D = 60–69.9, F < 60.

**Attendance Policy:** Attendance and participation in both lectures and lab sessions are expected. Per college policy, a student may miss up to 10% of class meetings without penalty (e.g. 3 absences in a 2/week course). Excessive absences or lateness may impact your grade. Hands-on activities and labs are difficult to make up, so please strive to attend and actively participate.

**Academic Integrity:** All work must be your own. Cheating or plagiarism (including uncredited copying of SQL scripts or code) is strictly prohibited. Collaborative discussion is encouraged, but each student must submit original work for individual assignments. Refer to the college’s Academic Integrity policy for details on consequences of violations.

## Course Schedule and Outline

**Week 1: Introduction to Database Administration**  
- **Topics:** Roles and tasks of a Database Administrator (DBA) in organizations; overview of DBA responsibilities (installation, maintenance, security, etc.)[[4]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Maintaining%20and%20troubleshooting%20databases)[[18]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Improving%20data%20security). Review of database design concepts: relational model, entity-relationship diagrams, normalization (to 3NF), and SQL query review (this refreshes prerequisite knowledge). Discussion of how good design underpins effective administration.  
- **Lab:** Course setup and environment orientation. Install PostgreSQL and pgAdmin (if not pre-installed). Verify students can create a connection to a local PostgreSQL server. Review an example database schema (provided by instructor) – use pgAdmin to explore tables, relationships, and run a few sample SELECT queries. This reinforces design review in a practical way.  
- **Readings:** *Watt,* *Database Design 2nd Ed.* **– Chapters 8 & 10 (ER Modeling and ERD) and Chapter 12 (Normalization) for design review**[**[19]**](https://opentextbc.ca/dbdesign01/#:~:text=Adrienne%20Watt)[**[20]**](https://opentextbc.ca/dbdesign01/#:~:text=11)**. Skim Chapter 14 (Database Users) to understand different roles of people interacting with DBs.** PostgreSQL Wikibook\*\* – “Introduction” chapter (for an overview of PostgreSQL and the scope of DB administration)[[1]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=PostgreSQL%20,a%20complete%20treatment%2C%20we%20recommend). Optionally, CompTIA article on DBA responsibilities[[4]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Maintaining%20and%20troubleshooting%20databases)[[18]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Improving%20data%20security) (for real-world context).

**Week 2: PostgreSQL Architecture and Setup**  
- **Topics:** **PostgreSQL Database Architecture** – Explore the client-server model and server internals[[14]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=PostgreSQL%20implements%20a%20client,at%20the%20server%20site). Components of PostgreSQL: database cluster vs. database vs. schema[[21]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20)[[22]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,RAM%2C%20and%20Files); physical structure (data files, WAL files, memory buffers) and processes (postmaster, background writer, WAL writer, autovacuum, etc.)[[23]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=At%20startup%20time%2C%20the%20Instance,processes%2C%20files%2C%20and%20their%20collaboration). Transaction processing basics in Postgres: ACID properties, commit and rollback, and how PostgreSQL’s MVCC (Multi-Version Concurrency Control) works to handle consistency and concurrency[[24]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=Clients%20do%20not%20have%20direct,PostgreSQL%20does%20not%20use%20threading)[[25]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91). Introduction to the PostgreSQL data dictionary (system catalogs) and utilities.  
- **Lab:** Initialize and configure a PostgreSQL instance. Use initdb or the installer to create a new database cluster. Modify basic settings in **postgresql.conf** (e.g. adjust memory settings lightly for demonstration) and **pg\_hba.conf** (to configure client authentication). Practice starting/stopping the PostgreSQL service. Use pgAdmin/psql to connect to the server and create a test database. Observe server logs when starting up and shutting down.  
- **Readings:** *PostgreSQL Wikibook* – “Architecture: Processes, RAM, Files”[[14]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=PostgreSQL%20implements%20a%20client,at%20the%20server%20site)[[23]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=At%20startup%20time%2C%20the%20Instance,processes%2C%20files%2C%20and%20their%20collaboration) and “Transactions & MVCC” sections[[25]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91) (to understand how Postgres handles concurrency). **PostgreSQL Manual** – Chapter “Overview of PostgreSQL Architecture” (for reference). Focus on sections explaining background processes and WAL (Write-Ahead Logging). These resources explain how Postgres manages memory and ensures durability[[23]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=At%20startup%20time%2C%20the%20Instance,processes%2C%20files%2C%20and%20their%20collaboration).

**Week 3: Managing Storage – Tablespaces and Data Files**  
- **Topics:** **Tablespaces and File Management in PostgreSQL:** Using tablespaces to control where data is stored on disk. Creating and managing tablespaces (for example, to put an index or user data on a different disk)[[21]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20). How PostgreSQL stores tables and indexes in the filesystem (relation files, OID naming). Discuss scenarios for using tablespaces (performance, space management). **Storage and Security:** File system permissions for database files; securing data files; understanding how PostgreSQL implements access controls at the OS level. **Data Loading and Unloading:** Introduction to bulk loading data using COPY command, and using external CSV files. Strategies for moving data between systems (mention that detailed data migration will be covered later in Week 12 with MongoDB).  
- **Lab:** Create a new tablespace on the local filesystem via SQL command and alter a sample table to use this tablespace. Verify the physical location of the data file. Practice a bulk load: using the COPY command to import a CSV dataset into a PostgreSQL table (the instructor will provide a sample dataset). Then use COPY TO or pgAdmin’s export to unload data to a file. Discuss any errors encountered and the need for proper formatting/permissions.  
- **Assessment:** **Quiz 1** (Week 3) – short quiz on Weeks 1–2 material (DBA roles, architecture basics).  
- **Readings:** *PostgreSQL Wikibook* – “The physical Perspective: Directories and Files” (covers how PostgreSQL organizes data files)[[21]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20). **Wikibook** – “Tablespace” section[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning) under Special Topics (explains what tablespaces are in Postgres). **PostgreSQL Manual** – Documentation on the COPY command (to assist with lab). Also read an online tutorial or blog on *PostgreSQL bulk data loading best practices* (open blog post).

**Week 4: Schema Management and SQL Objects**  
- **Topics:** **Schema Management in an RDBMS:** Review of SQL **DDL** (Data Definition Language) statements – CREATE/ALTER/DROP for schemas, tables, indexes, views, sequences, etc. Organizing database objects into schemas and managing schema privileges. **Managing Tables** – creating tables with appropriate data types; special table types (mention temporary tables). Partitioned tables basics (if applicable in Postgres). **Indexes** – when to create indexes, different index types in PostgreSQL (B-tree, Hash, GIN, etc.) – and how they affect performance. **Views** – creating views for security or convenience. **Sequences** – using sequences for auto-incrementing keys. **Constraints & Triggers** – ensuring data integrity with primary keys, foreign keys, unique constraints, check constraints; overview of triggers for automation. Students learn how these objects are created and managed by a DBA, and how to monitor them via system catalogs (e.g., querying information\_schema or pg\_catalog).  
- **Lab:** Hands-on SQL DDL. Using the test database, write and execute SQL to: create a new schema, transfer an existing table into that schema, and set appropriate privileges. Create a new table with constraints (primary key, foreign key referencing another table). Create a couple of indexes on table columns (and then use EXPLAIN to see the planner pick the index on a simple query). Create a view that subsets or joins data from tables. Create a sequence and link it to a table’s auto-increment field. If possible, demonstrate a simple trigger (e.g., an AFTER INSERT trigger that logs an audit record) – or discuss it conceptually if time is short.  
- **Readings:** *PostgreSQL Wikibook* – “Schema, Database, Cluster (logical perspective)”[[21]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20)[[22]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,RAM%2C%20and%20Files) and sections on **Indexes** and **Views** (consult the wikibook’s table of contents or PostgreSQL manual: e.g., read PostgreSQL Manual chapters on *Indexes* for an overview of index types). An open-source tutorial on **constraints and triggers** (for example, a blog or the PostgreSQL tutorial from postgres.org) can supplement understanding of these concepts. These readings will reinforce how a DBA manages various schema objects.

**Week 5: Managing Data and Concurrency**  
- **Topics:** **Transactions and Concurrency Control:** Deepen understanding of how PostgreSQL handles concurrent transactions. Locking in PostgreSQL – types of locks (row-level, table-level) and lock modes. The phenomena of concurrent access: dirty reads, phantom reads (review isolation levels: Read Committed vs. Repeatable Read, etc.). PostgreSQL’s MVCC implementation: how versioning and snapshots work to enable concurrent readers/writers[[23]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=At%20startup%20time%2C%20the%20Instance,processes%2C%20files%2C%20and%20their%20collaboration)[[25]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91). **UNDO/Redo and Vacuum:** Explain the concept of undo data/rollback segments (Oracle term) vs. how Postgres uses **WAL (Write-Ahead Log)** and **VACUUM** to clean up dead tuples[[26]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20)[[27]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91). The role of the autovacuum process in maintaining concurrency and space reuse. Introduce **pg\_stat\_activity** and other monitoring tools to view active sessions and locks.  
- **Lab:** Concurrency simulation. Using two separate connections (e.g., two psql sessions or pgAdmin query windows), execute transactions that update the same records to illustrate locking behavior. For example: Session A begins a transaction and updates a row without commit; Session B attempts to update the same row – observe it waiting. Then commit in A and see B proceed. Try different isolation levels (if using default Read Committed vs setting Repeatable Read in a session) and see effects (this can be done by reading some rows in one transaction and updating them in another). Run the pg\_stat\_activity view to identify the waiting query and lock. Also, demonstrate the effect of VACUUM: delete a large number of rows in a table, then run VACUUM VERBOSE and observe the output about reclaiming space (or auto-vacuum if enabled).  
- **Assessment:** **Quiz 2** (Week 5) – covers Weeks 3–5 content (storage, schema objects, transactions).  
- **Readings:** *PostgreSQL Wikibook* – “Transactions” and “Multiversion Concurrency Control (MVCC)” sections[[28]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=The%20ACID%20Paradigm%20%20,Implementation%20of%20the%20ACID%20Paradigm)[[27]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91), and the “Vacuum” section[[29]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Paradigm). These explain how PostgreSQL manages concurrent access and why vacuuming is needed. Optionally, read the PostgreSQL documentation on **Explicit Locking** to understand lock commands and the system view pg\_locks. Also recommended: a tutorial or blog on *handling transactions in PostgreSQL* (open content, e.g., from postgresguide or DigitalOcean).

**Week 6: Database Installation, Upgrades, and Maintenance Tasks**  
- **Topics:** **DBMS Installation & Configuration Management:** Considerations in installing a DBMS in production (init parameters, file locations). Scripting installations or using package managers. **Upgrading PostgreSQL:** Overview of minor vs. major upgrades; using pg\_dump/pg\_restore or the pg\_upgrade tool for major version upgrades. Discuss planning an upgrade to minimize downtime. **Routine Maintenance:** Rebuilding indexes (when necessary), analyzing statistics (using ANALYZE to update optimizer stats), and monitoring disk usage. **Uninstalling or Moving** a database: considerations for properly shutting down and moving data files if needed. **Basic Troubleshooting:** Interpreting log files, common errors on startup (e.g., port already in use, data directory permissions issues) and how to address them.  
- **Lab:** Perform a simulated “upgrade”: if possible, the instructor provides an older version of PostgreSQL on a test dataset. Students practice using pg\_dumpall or pg\_dump to export a database, then import into a newer version instance (this can be done on the same machine if two versions are available, or just conceptually discuss). Alternatively, use the same version to simulate migration: dump the database, drop it, and restore it – to go through the motions of backup/restore (serves as practice for backup which is coming next week). Also, use EXPLAIN ANALYZE on various queries before and after running ANALYZE to see the effect of updated statistics on query plans (this demonstrates a maintenance task). If time permits, deliberately mis-configure something (e.g. set an invalid parameter in postgresql.conf) and attempt to restart to see error, then fix it.  
- **Readings:** *PostgreSQL Wikibook* – “Configuration”[[15]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,Backup%20%26%20Recovery) and “Upgrade” sections[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning). PostgreSQL Manual: chapter on **Routine Database Maintenance Tasks** (covers VACUUM, ANALYZE, index rebuild, etc.). Also, a short guide on **pg\_upgrade** from the official docs or wiki (outlining steps to upgrade PostgreSQL). These readings will prepare students for maintaining and upgrading DBMS software.

**Week 7:** Review and Midterm Exam***\****  
***-* \*Topics:** In-class review of key concepts from Weeks 1–6. We will summarize relational DB administration tasks covered: architecture, storage management, schema management, concurrency, maintenance. Address any questions students have, possibly with a quick look at how these concepts compare to Oracle (since the original course was Oracle-based) to ensure broad understanding (e.g., terminology differences like Oracle “UNDO” vs. Postgres MVCC, etc.).  
- **Midterm Exam:** Conducted during class (written and/or practical). It will test theoretical understanding (e.g. explain what the WAL does, how to create a role) as well as practical skills (interpret an **EXPLAIN** output, write SQL DDL for a scenario, or debug a provided SQL script). The exam covers all PostgreSQL administration topics from Weeks 1–6.  
- *No new lab this week.* After the exam, the second half of class (if time remains) will introduce the course project, especially as it relates to both technologies: students receive the project description and we discuss expectations. Groups can be formed if applicable.

**Week 8: Introduction to MongoDB Atlas and NoSQL**  
- **Topics:** **NoSQL and MongoDB Concepts:** Rationale for NoSQL databases – differences from relational (flexible schema, horizontal scaling, etc.). Introduction to MongoDB: JSON/BSON document data model, collections in place of tables, no fixed schema enforcement[[30]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20in%20MongoDB%20has%20a,hold%20different%20types%20of%20data). **MongoDB Atlas**: using cloud-managed MongoDB – benefits (managed infrastructure, high availability by default). **Setting up MongoDB Atlas:** Walk through creating an Atlas account, deploying a free-tier cluster, and understanding cluster settings (regions, cluster tier, etc.). **Connecting to Atlas:** how to obtain a connection string and connect via MongoDB Compass or the Mongo shell. Basic CRUD in MongoDB conceptually (we will do hands-on next week, but introduce the idea of insert/find/update/delete in a document store). Emphasize that the next labs will be done in the cloud environment.  
- **Lab:** **Atlas Setup Lab:** Each student (or team) creates a MongoDB Atlas free-tier cluster. Guide them through: signing up, creating a project, provisioning a cluster (M0 sandbox). Then configure network access (whitelist their IP or use 0.0.0.0 for all, as appropriate) and create a database user with password. Using **MongoDB Compass** (or the Atlas web GUI), connect to the new cluster. Verify the connection by viewing the default databases or creating a new database and collection via the GUI. Load sample data: Atlas provides an option to load a sample dataset (e.g., sample Airbnb listings or similar) – if available, have students add a sample dataset to their cluster for use in upcoming exercises. Ensure everyone has a functional cluster for subsequent weeks.  
- **Readings:** **MongoDB Atlas Documentation** – *Getting Started with Atlas* (step-by-step guide to deploy a free cluster)[[31]](https://www.mongodb.com/resources/products/platform/mongodb-atlas-tutorial#:~:text=How%20to%20use%20MongoDB%20Atlas%3F,and%20a%20user%2C%20and). *Tutorialspoint MongoDB* – “MongoDB Environment” (covers setting up the environment, though focused on local install, it’s useful to skim) and **“MongoDB Atlas Tutorial”** (if available as an article) to reinforce the steps. Also *MongoDB Manual* – “Introduction to MongoDB” page for an overview of architecture (e.g., how MongoDB is distributed by design, role of config servers in advanced cases, etc.). These will help students unfamiliar with NoSQL to grasp the basics.

**Week 9: MongoDB Data Modeling and CRUD Operations**  
- **Topics:** **Data Modeling in MongoDB:** Understanding schema-less design. Difference between embedding vs. referencing data in MongoDB[[32]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=MongoDB%20provides%20two%20types%20of,models%20while%20preparing%20your%20document)[[33]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Normalized%20Data%20Model). Pros and cons of **embedded documents** (denormalized, all data in one document) vs. **normalized references** (like foreign keys, storing references to data in other collections)[[6]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Embedded%20Data%20Model)[[33]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Normalized%20Data%20Model). Guidelines for modeling: data that is accessed together should be stored together (embed), and when to use references (to avoid duplication or for many-to-many). Discuss example: modeling a blog with posts and comments – show two approaches (embedding comments inside post vs. separate collection)[[34]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Example)[[35]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=,be%20zero%20or%20more%20comments). **CRUD Operations in MongoDB:** Using MongoDB’s API for Create, Read, Update, Delete. Methods: insertOne, insertMany; find (with query filters); updateOne/updateMany and update operators (e.g., $set, $push); deleteOne/deleteMany. How MongoDB’s query language differs from SQL (JSON-based queries vs. SELECT syntax). Introduce the concept of **BSON** and supported data types in Mongo (e.g., ObjectId, arrays, sub-documents).  
- **Lab:** Using **MongoDB Compass** or the shell to perform CRUD. In the Atlas cluster (with sample data or a new test collection):  
- Perform **Create:** Insert a few sample documents into a new collection (e.g., a students or products collection) using Compass’s GUI or db.collection.insertOne() in the shell.  
- **Read:** Query the collection with simple filters (e.g., find all documents where field X = value). Try more complex queries with conditions and projection (selecting certain fields) using Compass query bar or shell queries.  
- **Update:** Use updateOne to modify a field in documents (e.g., increment a counter, or add a new field to documents). Use $set and $unset. Observe the result by querying again.  
- **Delete:** Remove a document or subset of documents with deleteOne/deleteMany (with a query filter). Emphasize caution as MongoDB will delete entire documents matching criteria.  
Students should also practice using the Atlas web UI: Atlas provides a Data Explorer where they can insert and query documents without installing anything, which can be shown as an alternative.  
- **Readings:** *Tutorialspoint MongoDB* – “MongoDB Data Modeling”[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design)[[6]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Embedded%20Data%20Model) (for embedding vs. normalization, with examples) and “MongoDB CRUD Operations” (chapters “Insert Document”[[36]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Sorting%20Records), “Query Document”, “Update Document”, “Delete Document”). **MongoDB Manual** – “CRUD Operations” section[[17]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=MongoDB%20is%20a%20persistent%20document,fundamental%20types%20of%20data%20operations)[[37]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=These%20four%20operations%20are%20jointly,referred%20to%20as%20CRUD%20operations) for a more thorough explanation (including examples of queries and updates). Optionally, *DigitalOcean Tutorial* on CRUD in MongoDB[[17]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=MongoDB%20is%20a%20persistent%20document,fundamental%20types%20of%20data%20operations)[[37]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=These%20four%20operations%20are%20jointly,referred%20to%20as%20CRUD%20operations) – which explains these operations in a beginner-friendly way.

**Week 10: Indexing and Performance Tuning in MongoDB**  
- **Topics:** **Indexes in MongoDB:** What an index is in a document database and how it improves query performance by allowing efficient data lookup[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index). Creating indexes on fields in MongoDB (single field index, compound indexes). **Index types:** brief mention of default \_id index, unique indexes, and text indexes (if applicable). **Using Indexes:** How to examine whether queries use indexes – introduction to explain() in MongoDB (similar to SQL EXPLAIN). Understand the concept of collection scan vs. indexed find. **Performance Tuning:** Basic strategies in MongoDB – proper indexing, avoiding large document size where not needed, using projections to limit returned fields, and leveraging MongoDB’s aggregation framework for data processing on the server side. Also discuss how MongoDB handles **load**: it’s distributed, but within a single node, considerations like locking at document level (Mongo uses document-level concurrency). Mention the role of memory (the WiredTiger cache) and importance of monitoring metrics like cache hit ratio.  
- **Lab:** **Indexing Lab:** Using the Atlas sample dataset (e.g., if using the **sample\_mflix** movie database or similar), identify a query pattern (for example, search by a field that is not indexed). Execute a query and measure its time or observe it in the Atlas performance panel (if available). Then create an index on that field: e.g., db.movies.createIndex({ "genre": 1 }). Re-run the same query and observe the improvement using explain("executionStats") in the shell or Compass Explain Plan feature – confirm that the index is being used (index scan vs collection scan) and note the difference in reported execution time or keys examined. Try a compound index if relevant (e.g., index on two fields commonly queried together). Also demonstrate a **text index** on a string field and using a text search ($text query) if time permits.  
Additionally, students will use the **Monitoring** tools in Atlas: navigate to the Metrics charts for the cluster to see operations count, CPU utilization, etc., especially while running a heavy query vs. an indexed query (to connect how indexing reduces load).  
- **Assessment:** **Quiz 3** (Week 10) – covers Weeks 8–10 (Atlas setup, MongoDB modeling, CRUD, indexing concepts).  
- **Readings:** *Tutorialspoint MongoDB* – “MongoDB Indexing” chapter[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index) (defines indexes and how to create them) and “Covered Queries”/“Analyzing Queries”[[38]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Map%20Reduce)[[39]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Regular%20Expression) (if available, describing how to interpret explain output). **MongoDB Manual** – “Indexes” introduction and “Explain Results” (sections that show how to use explain in MongoDB). Also, a MongoDB blog post or documentation page on **performance best practices** (covering proper indexing, not scanning large collections, etc.) – for example, *MongoDB’s guide to optimization* (freely available on their blog).

**Week 11: Data Migration and Import/Export (SQL ↔ NoSQL)**  
- **Topics:** **Data Migration Concepts:** Strategies for moving data between relational and NoSQL systems. Discuss use cases: maybe migrating a subset of data from PostgreSQL to MongoDB for a specific application, or vice versa. **Tools for Data Import/Export:** Using **mongoimport** and **mongoexport** for JSON/CSV data. Using PostgreSQL’s COPY or CSV export to get data out, then transforming to JSON. Challenges in migration: differences in schema (e.g., how to represent relational rows as JSON documents), data type compatibility. **ETL basics:** We won’t build a full ETL pipeline, but mention how a script or intermediary (Python, etc.) could be used to transform data. Also cover importing data into MongoDB Atlas via Compass (Compass can import JSON/CSV) or the Atlas Data Import tool. **MongoDB Aggregation Framework:** Introduction to the aggregation pipeline as a way to transform data within MongoDB (for example, to reshape documents or summarize data). Not in depth, but enough to mention it exists for data transformation needs.  
- **Lab:** Perform a simple migration exercise: Export a small dataset from PostgreSQL and import into MongoDB. For example, from our earlier Postgres labs, take a table (e.g., a customer list or products list). Use COPY TO CSV in Postgres (or pgAdmin’s export function) to get data to a CSV file. Then use **MongoDB Compass** or mongoimport to import that CSV into a new MongoDB collection. Check that the documents are created and discuss how each row became a JSON document (flat structure by default). If needed, refine the approach: perhaps tweak the CSV or use a JSON export from Postgres (with JSON functions) for a richer nested structure. Conversely, demonstrate exporting data from MongoDB: use mongoexport to get a JSON or CSV dump of a Mongo collection, and if feasible, use PostgreSQL foreign data wrapper or copy command to bring it into a Postgres table (this might be more complex, so could be just described if not done). The main goal is to show students how to move data around between systems and formats.  
- **Readings:** Blog or tutorial on **Importing Data to MongoDB Atlas** (e.g., MongoDB documentation page “Import and Export Data” which covers mongoimport/mongoexport)[[40]](https://studio3t.com/knowledge-base/articles/mongodb-atlas-tutorial/#:~:text=MongoDB%20Atlas%20Tutorial%3A%20How%20to,in%20this%20MongoDB%20Atlas%20tutorial). *Tutorialspoint MongoDB* – “MongoDB - Export/Import” (if such a chapter exists under “Backup” or similar) – for example, the **Create Backup** page partially covers backup but not the import; however, using mongodump/mongorestore is analogous[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server). Also, refer to **PostgreSQL Docs** on importing/exporting CSV (for the Postgres side of things). NoSQL-to-SQL migration article (optional, if an open resource is available, e.g., MongoDB blog on using their BI connector, but that might be too specific).

**Week 12: MongoDB Security and User Management**  
- **Topics:** **MongoDB Security Basics:** Differences from SQL in how authentication and authorization are handled. **Atlas Security Features:** Network access control (IP whitelisting or VPC peering), encryption (at rest and in transit – Atlas has TLS by default). **User Roles in MongoDB:** Creating users in MongoDB with specific roles (readWrite, dbAdmin, etc.)[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning). How MongoDB’s roles map to privileges on databases/collections (similar in concept to SQL GRANTs). The concept of the **admin** database in MongoDB for server-wide users vs. per-database users. **Monitoring & Auditing:** Overview of how to monitor database access – e.g., MongoDB Atlas provides monitoring and can log operations. Mention **MongoDB Ops Manager**/Cloud Manager just briefly as related to enterprise, but focus on Atlas built-in tools. **Basic Hardening:** Ensure students understand to never leave an open database (in Atlas we have defaults, but on-prem Mongo without auth is a classic mistake). Also, discuss SQL vs. NoSQL security: in SQL (Postgres) we did roles and GRANTs; in Mongo, we create users with roles – conceptually similar end goal of controlling access.  
- **Lab:** **User Management Lab:** Using the Atlas UI or Mongo shell to create a new database user with restricted privileges. For instance, create a user that only has readWrite on a specific database in the cluster. Then test that user’s access by connecting (perhaps using the shell with that username) and attempting allowed operations vs. forbidden ones (e.g., try to write to a different database – it should fail). Also, explore the **Atlas security settings**: view the IP access list (and add/remove an entry), ensure connections require password. If available in Atlas free tier, demonstrate enabling 2-factor auth or integrating with some identity provider (though that might not apply to free tier; it’s more enterprise, so just mention it). Finally, utilize the **Activity Feed** or monitoring to see any suspicious login attempts (if any, or just describe it).  
- **Readings:** *MongoDB Manual* – “Security Introduction” and “Enable Authentication” (these explain creating users and roles). *Tutorialspoint MongoDB* – “MongoDB Security” or the sections on user management (not explicitly seen in ToC, but some info might be under “Database Administration” topics). Additionally, MongoDB Atlas documentation on **Network Security** (IP access lists) and **Database Users** in Atlas (which is a GUI way but corresponds to underlying MongoDB users). Compare with a brief reading of PostgreSQL’s security (pg\_hba.conf and roles) to solidify understanding of differences. This will provide context that security is critical in both systems, implemented differently.

**Week 13: Backup, Recovery and Monitoring in MongoDB**  
- **Topics:** **Backup and Restore in MongoDB:** Using mongodump and mongorestore for full backups[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server). Atlas backups: explain that Atlas has an automated backup system even for free tier (with certain limitations) – show how to take a snapshot in Atlas if possible. Discuss point-in-time recovery concept (for higher clusters with oplog). **Recovery**: how to restore data from dump files. Also discuss **mongoexport/mongoimport** for selective backup of collections to JSON/CSV. **Monitoring & Diagnostics:** Using MongoDB Atlas monitoring tools – overview of metrics like operation executions, slow query log (Atlas has Performance Advisor that suggests indexes – mention this feature). Using the **Atlas Alerts** (setting alerts for high CPU, etc.). **Distributed considerations:** Basic introduction to replica sets (MongoDB’s primary-secondary replication) and how having replication enables backup without downtime and automatic failover[[41]](https://www.openmymind.net/mongodb.pdf#:~:text=Replication%20MongoDB%20replication%20works%20in,Writes%20are)[[42]](https://www.openmymind.net/mongodb.pdf#:~:text=While%20replication%20can%20help%20performance,Sharding%20is%20the%20primary). Also mention sharded clusters (though we will cover next week in detail) but in context that backing up a sharded cluster is more involved (needs config server backup, etc., beyond scope). The goal is to ensure students know how to not lose data and how to observe the health of their Mongo deployment.  
- **Lab:** **Backup/Restore Lab:** Perform a backup of a MongoDB database and restore it. For Atlas free tier, since direct filesystem access isn’t available, use **mongodump** via the command line, connecting to the Atlas cluster (the connection string can be used with mongodump). Dump one of the databases (perhaps the one they created for lab exercises). Then deliberately drop a collection or document in that database to simulate data loss. Next, use **mongorestore** to restore from the dump, and verify the data is back. (Alternatively, if using Compass only, Compass has an “Import Collection” that could serve a similar function for a single collection from a JSON export.) Also, test exporting data: use mongoexport to export a collection to JSON or CSV (similar to what was done in Week 11, reinforcing that skill). For monitoring, have students open the Atlas Metrics and identify a period of activity (during their backup perhaps) to see how it reflects. If possible, enable an alert (Atlas lets you set alerts on certain conditions) – e.g., set an alert for “connections > X” and then simulate it by opening multiple connections. This may be limited in free tier, but you can at least show the interface.  
- **Assessment:** **Quiz 4** (Week 13) – covers Weeks 11–13 (migration, security, backup/monitoring in MongoDB).  
- **Readings:** *Tutorialspoint MongoDB* – “MongoDB Create Backup”[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server) and “MongoDB Deployment” (which likely touches on replication/sharding) sections. **MongoDB Manual** – “Backup and Restoration Methods” and specifically the usage of mongodump/mongorestore (with examples). Atlas documentation on backups (note: free tier backup vs paid tier point-in-time) – just for awareness. Also, a short piece on **MongoDB Replication** (since we will dive more next week, reading now will prepare them)[[41]](https://www.openmymind.net/mongodb.pdf#:~:text=Replication%20MongoDB%20replication%20works%20in,Writes%20are)[[42]](https://www.openmymind.net/mongodb.pdf#:~:text=While%20replication%20can%20help%20performance,Sharding%20is%20the%20primary).

**Week 14: Advanced Topics – Distributed Databases (Replication & Sharding) and Course Project Presentations**  
- **Topics:** **Distributed Database Management:** Explore how large databases scale out. **MongoDB Replication:** Detailed look at how MongoDB replica sets work (primary-secondary architecture, replication oplog, eventual consistency considerations on secondaries)[[41]](https://www.openmymind.net/mongodb.pdf#:~:text=Replication%20MongoDB%20replication%20works%20in,Writes%20are)[[42]](https://www.openmymind.net/mongodb.pdf#:~:text=While%20replication%20can%20help%20performance,Sharding%20is%20the%20primary). How to configure a replica set (conceptually, since Atlas auto-configures replication). Failover process in MongoDB – what happens when a primary goes down. **MongoDB Sharding:** Basics of sharding (horizontal partitioning across multiple nodes). Role of the config server and query router (mongos). Choosing a shard key and how data is distributed. We will **not** implement a sharded cluster in lab (not possible on free tier), but understanding the concept is important for a DBA working with big data. **Comparative glance at Distributed SQL:** (very brief) – mention that PostgreSQL can scale reads via replication (streaming replicas) and there are extensions for sharding (Postgres-XC, etc.), but these are advanced; the main idea is to show relational vs. NoSQL approaches to scaling. **Course Wrap-Up:** Summarize key learnings from both halves of the course. Discuss emerging trends (cloud DB services, automation, DBA in DevOps, etc.).  
- **Project Presentations:** Students (or teams) present their case study/project. Each presentation should outline the problem scenario, their database design and setup, key administration tasks they performed, and lessons learned. For example, a project might involve designing a small system that uses PostgreSQL for transactional data and MongoDB for logging or analytics, with the student demonstrating how they managed both. Students should describe how they handled user management, performance tuning, backups, etc., in their project. This is an opportunity to articulate and solidify the practical skills gained.  
- **Lab:** No new technical lab, but class time is devoted to **Project Demo/Review**. Instructor and peers may ask questions about how a group implemented certain admin tasks. This simulates a real-world scenario of DBAs discussing their systems.  
- **Readings:** *Tutorialspoint MongoDB* – “Replication” and “Sharding” chapters for conceptual understanding (if not already read)[[43]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=,be%20zero%20or%20more%20comments)[[41]](https://www.openmymind.net/mongodb.pdf#:~:text=Replication%20MongoDB%20replication%20works%20in,Writes%20are). Also, skim MongoDB Manual pages on replication (especially the replication mechanics and how to add a member to a replica set) and sharding (focusing on the idea of a shard key). No new readings beyond this, as focus is on project work. If students are interested, references to external case studies of large-scale MongoDB or PostgreSQL deployments can be given as optional inspiration.

**Week 15: Final Exam**  
- The **Final Exam** will be administered covering material from Weeks 8–14 (NoSQL/MongoDB section and any comparative concepts). Students should be prepared for questions on MongoDB architecture, CRUD, indexing, security, backup, and distributed setup concepts. There may also be synthesis questions that involve comparing relational and NoSQL administration or designing a solution involving both.  
- After the exam, we will have a course debrief and students can provide feedback on the open-source materials and tools used. We will also discuss next steps for those continuing in data management (e.g., advanced database courses, certifications like MongoDB DBA or Postgres certifications, etc.).  
- *No lab –* the final exam will likely occupy the session. Students should ensure all project deliverables are submitted as well (if the presentation was in Week 14, any final report is due by Week 15).

## Term Project Overview (Case Study)

Throughout the course, students will work on a capstone project that integrates both PostgreSQL and MongoDB administration skills. The project will be introduced mid-semester (after the PostgreSQL portion) and can be done in small teams or individually. Possible project scenarios include: - **Hybrid Data Store Design:** For example, a web application with an SQL database for core transactional data (users, orders) and a NoSQL database for storing logs or user activity feeds. Students act as DBAs to set up both databases, configure them, and optimize them for the use case.  
- **Migration Case Study:** Take an existing dataset and move part of it from SQL to NoSQL, justifying the move (e.g., handling unstructured data). Implement the migration and compare query performance on both systems.  
- **Admin Automation:** Set up scripts or use tools to automate backup and monitoring for both a Postgres and a MongoDB instance, documenting the process and scheduling.

**Project Deliverables:** A short proposal, an interim progress report, and a final report + presentation. The final submission should include: - Description of the scenario and requirements.  
- Design of the database schema (ER diagram for SQL, and document schema examples for MongoDB).  
- Description of setup steps (how the PostgreSQL database was created, configurations set; how the MongoDB cluster was set up on Atlas, any custom settings).  
- Demonstration of key admin tasks: user creation, example queries and their optimization (EXPLAIN plan from Postgres, index used in Mongo, etc.), backup strategy implemented (with evidence, e.g., screenshots or logs of backup tools), and security measures (like sample role definitions).  
- Reflections on challenges faced (e.g., schema translation issues, performance tuning difficulties) and how they were resolved.

This project is an opportunity to apply course concepts in a realistic scenario and will be graded on correctness, completeness, use of open-source tools, and clarity of explanation. Creativity in scenario or use of additional open-source automation/monitoring tools will earn bonus points (for instance, using a configuration management script, or exploring an extension like pgAudit for PostgreSQL auditing).

**Note:** All resources used in the project must be free or open-source. Students are encouraged to use official documentation[[16]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,which%20summarizes%20some%20special%20topics) and community forums for help, but must cite references for any significant external solutions incorporated.

**End of Syllabus** – This schedule is subject to adjustment based on class progress. All changes and additional resources will be communicated in advance. Let’s have a productive semester learning database administration with modern, open-source technologies!

Sources: [[1]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=PostgreSQL%20,a%20complete%20treatment%2C%20we%20recommend)[[2]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Database%20administrators%20are%20becoming%20more,got%20the%20answers%20you%20need)[[14]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=PostgreSQL%20implements%20a%20client,at%20the%20server%20site)[[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning)[[15]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,Backup%20%26%20Recovery)[[44]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=MongoDB%20provides%20two%20types%20of,models%20while%20preparing%20your%20document)[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index)[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server)[[17]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=MongoDB%20is%20a%20persistent%20document,fundamental%20types%20of%20data%20operations)

[[1]](https://en.wikibooks.org/wiki/PostgreSQL" \l ":~:text=PostgreSQL%20,a%20complete%20treatment%2C%20we%20recommend) [[7]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%2A%20Client,Partitioning) [[15]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,Backup%20%26%20Recovery) [[16]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,which%20summarizes%20some%20special%20topics) [[21]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20) [[22]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=,RAM%2C%20and%20Files) [[25]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91) [[26]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=%5Bedit%20%20) [[27]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Visibility%20of%20Rows%3A%20Isolation%20Levels,91) [[28]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=The%20ACID%20Paradigm%20%20,Implementation%20of%20the%20ACID%20Paradigm) [[29]](https://en.wikibooks.org/wiki/PostgreSQL#:~:text=Paradigm) PostgreSQL - Wikibooks, open books for an open world

<https://en.wikibooks.org/wiki/PostgreSQL>

[[2]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Database%20administrators%20are%20becoming%20more,got%20the%20answers%20you%20need) [[3]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Key%20responsibilities%20of%20a%20database,administrator) [[4]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Maintaining%20and%20troubleshooting%20databases) [[18]](https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/#:~:text=Improving%20data%20security) What Does a Database Administrator Do?

<https://www.comptia.org/en-us/blog/what-does-a-database-administrator-do/>

[[5]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20Model%20Design) [[6]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Embedded%20Data%20Model) [[8]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Considerations%20while%20designing%20Schema%20in,MongoDB) [[30]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Data%20in%20MongoDB%20has%20a,hold%20different%20types%20of%20data) [[32]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=MongoDB%20provides%20two%20types%20of,models%20while%20preparing%20your%20document) [[33]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Normalized%20Data%20Model) [[34]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=Example) [[35]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=,be%20zero%20or%20more%20comments) [[43]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=,be%20zero%20or%20more%20comments) [[44]](https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm#:~:text=MongoDB%20provides%20two%20types%20of,models%20while%20preparing%20your%20document) MongoDB Data Modeling

<https://www.tutorialspoint.com/mongodb/mongodb_data_modeling.htm>

[[9]](https://thoughtbot.com/blog/reading-an-explain-analyze-query-plan#:~:text=The%20most%20powerful%20tool%20at,SQL%20queries%20is%20EXPLAIN%20ANALYZE) Reading a Postgres EXPLAIN ANALYZE Query Plan

<https://thoughtbot.com/blog/reading-an-explain-analyze-query-plan>

[[10]](https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm#:~:text=To%20create%20backup%20of%20database,backup%20of%20your%20remote%20server) MongoDB Backup Creation

<https://www.tutorialspoint.com/mongodb/mongodb_create_backup.htm>

[[11]](https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm#:~:text=Indexes%20are%20special%20data%20structures%2C,as%20specified%20in%20the%20index) MongoDB Indexing

<https://www.tutorialspoint.com/mongodb/mongodb_indexing.htm>

[[12]](https://opentextbc.ca/dbdesign01/#:~:text=Book%20Description) [[13]](https://opentextbc.ca/dbdesign01/#:~:text=5,Database%20development%20process) [[19]](https://opentextbc.ca/dbdesign01/#:~:text=Adrienne%20Watt) [[20]](https://opentextbc.ca/dbdesign01/#:~:text=11) Database Design – 2nd Edition – The BC Open Textbook Project

<https://opentextbc.ca/dbdesign01/>

[[14]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=PostgreSQL%20implements%20a%20client,at%20the%20server%20site) [[23]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=At%20startup%20time%2C%20the%20Instance,processes%2C%20files%2C%20and%20their%20collaboration) [[24]](https://en.wikibooks.org/wiki/PostgreSQL/Architecture#:~:text=Clients%20do%20not%20have%20direct,PostgreSQL%20does%20not%20use%20threading) PostgreSQL/Architecture - Wikibooks, open books for an open world

<https://en.wikibooks.org/wiki/PostgreSQL/Architecture>

[[17]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=MongoDB%20is%20a%20persistent%20document,fundamental%20types%20of%20data%20operations) [[37]](https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb#:~:text=These%20four%20operations%20are%20jointly,referred%20to%20as%20CRUD%20operations) How To Perform CRUD Operations in MongoDB | DigitalOcean

<https://www.digitalocean.com/community/tutorials/how-to-perform-crud-operations-in-mongodb>

[[31]](https://www.mongodb.com/resources/products/platform/mongodb-atlas-tutorial#:~:text=How%20to%20use%20MongoDB%20Atlas%3F,and%20a%20user%2C%20and) MongoDB Atlas Tutorial

<https://www.mongodb.com/resources/products/platform/mongodb-atlas-tutorial>

[[36]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Sorting%20Records) [[38]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Map%20Reduce) [[39]](https://www.tutorialspoint.com/mongodb/index.htm#:~:text=%2A%20MongoDB%20,Regular%20Expression) MongoDB Tutorial

<https://www.tutorialspoint.com/mongodb/index.htm>

[[40]](https://studio3t.com/knowledge-base/articles/mongodb-atlas-tutorial/#:~:text=MongoDB%20Atlas%20Tutorial%3A%20How%20to,in%20this%20MongoDB%20Atlas%20tutorial) MongoDB Atlas Tutorial: How to Set Up a Free Cluster & Import Data

<https://studio3t.com/knowledge-base/articles/mongodb-atlas-tutorial/>

[[41]](https://www.openmymind.net/mongodb.pdf#:~:text=Replication%20MongoDB%20replication%20works%20in,Writes%20are) [[42]](https://www.openmymind.net/mongodb.pdf#:~:text=While%20replication%20can%20help%20performance,Sharding%20is%20the%20primary) openmymind.net

<https://www.openmymind.net/mongodb.pdf>