Course Introduction

COSC 516 – Cloud Databases

Introductions – Dr. Ramon Lawrence

Professor, Computer Science

Research area: database systems, Internet of Things, software development

Teaching experience:

- 2020 Killam Teaching Prize (top teaching award at UBCO – one per year)
- 2017 UBCO Teaching Excellence Award Winner (two per year)
- 9-time member of teaching honour roll (top 10% instructors)

Industry experience: GE Big Data, UnityJDBC company/consulting

Note: May address me as "Dr. Lawrence", "Professor", or "Ramon" (pronounced RAY-MUN).

The Essence of the Course

The overall goal of this course is for you to:

Understand how cloud databases work and be proficient in their use:

1) Deploy databases on a variety of cloud vendor platforms

2) Decide when to use particular systems based on their technical characteristics, costs, and deployment features

3) Communicate effectively with other technical professionals when making design and deployment decisions and using new systems

This course provides valuable, applied training that is highly desired by employers.

My Course Goals

1) Provide the information in an effective way for learning.
2) Inspire and motivate students to learn and appreciate benefits of the course.
3) Strive for ***all*** students to understand the material and excel.
4) Be available for questions during scheduled times, office hours, and at other times as needed.
5) Teach students how to be a sophisticated cloud database user by understanding the key features of various systems provided by cloud vendors.

Your Course Goals

1) Sufficiently learn the material to pass the course.
2) Develop experience in using a variety of cloud database systems.
3) Learn how different database systems work in order to better understand how to use them properly.
4) Develop proficiency in using Amazon Web Services, Microsoft Azure, and Google Cloud.
5) Practice technical communication (verbal and written) for communicating with other professionals.
6) Determine if you want to perform database related research.

Course Objectives

1) Understand different database systems and their particular use cases.
2) Proficiency in creating a database system on each of the three major cloud providers (AWS, Azure, Google).
3) Experience creating technical summary documents and learning activities to educate others.
4) Mastery of constructing programs to interact with cloud databases including using testing frameworks.
5) Exposure to a variety of database technologies and systems and the ability to determine when to use them.
6) Practice allocating and managing resources on a cloud provider and monitoring billing and costs.

Academic Dishonesty

Cheating is strictly prohibited and is taken very seriously by UBC.

A guideline to what constitutes cheating:

- Labs
  - Submitting code produced by others.
  - Working in groups to solve questions and/or comparing answers to questions once they have been solved (except for group assignments).
  - Discussing HOW to solve a particular question instead of WHAT the question involves.
- Exams
  - Only materials permitted by instructor should be used in an exam.

Academic dishonesty may result in a "F" for the course and further actions by the Dean's office.

How to Excel in This Course

Attend ***every*** class:

- Review materials ***before*** class as preparation and complete any questions.
- Participate in class exercises and questions.

Complete all lab assignments:

- Labs practice the fundamental employable skills as well as being for marks.

Practice on your own. Practice makes perfect.

- Do more questions than in the labs.
- Read the additional reference material and spend more time working with each of the systems.

Database Project

The project requires selecting a particular cloud database and producing the following materials describing it:

- Summary PowerPoint presentation of its key features, benefits, and costs.
- Narrated video of presentation.
- Video demonstrating how to perform basic setup on the cloud provider.
- An auto-graded lab assignment for other students to do to learn how to use the system. These labs are for marks for the course.

Timeline:

- The first four weeks will have example systems presented by the instructor in the same format as required for the project.
- Starting in mid-October students will present their system, one per week. The project team is responsible for presenting the system in the first class of the week and supporting the lab assignment in the second class of the week.

The project is done in groups of **two or three**.

The Lab Assignments

The lab assignments are worth ***40%*** of your overall grade.

You have until the end of each week (Sunday midnight) to complete the lab for that week.

- No late assignments will be accepted.
- An assignment may be handed in any time before the due date and may be marked immediately.
- Lab assignments may take between 2 and 10 hours depending on the lab.

Lab assignments are done in ***project teams*** (two or three people).

My Expectations

My goal is for you to learn the material and walk out of this course confident in your abilities:

- To understand how cloud databases work and deploy/configure them properly
- To make intelligent decisions on when to use different database systems
- To be fluent in using cloud provider systems and understanding costs/benefits
- To communicate technical information effectively and professionally

I have high standards on the amount and difficulty of material that we cover. I expect a strong, continual effort in keeping up with material and participating in class activities.

The course is straightforward – if you do the work, you will do well.

Your mark is 50% perspiration and 50% inspiration.

What is a database?

A ***database*** is a collection of logically related data for a particular domain.

A ***database management system*** (***DBMS***) is software designed for the creation and management of databases.

- e.g. Oracle, DB2, Microsoft Access, MySQL, SQL Server, MongoDB

Bottom line: A ***database*** is the *data* stored and a ***database system*** is the *software* that manages the data.

Databases in the Real-World

Databases are everywhere in the real-world even though you do not often interact with them directly.

- $50 billion dollar annual industry

Examples:

- Retailers manage their products and sales using a database.
  - Wal-Mart has one of the largest databases in the world!
- Online web sites such as Amazon, eBay, and Expedia track orders, shipments, and customers using databases.
- The university maintains all your registration information and marks in a database that is accessible over the Internet.

Can you think of other examples?

What data do you have?

Database System Properties

A database system provides *efficient*, *convenient*, and *safe multi-user* storage and access to *massive* amounts of *persistent* data.

Efficient - Able to handle large data sets and complex queries without searching all files and data items.

Convenient - Easy to write queries to retrieve data.

Safe - Protects data from system failures and hackers.

Massive - Database sizes in gigabytes, terabytes and petabytes.

Persistent - Data exists even if have a failure.

Multi-user - More than one user can access and update data at the same time while preserving consistency.

Database Terminology

A ***data model*** is a collection of concepts that is used to describe the structure of a database. E.g. relational, XML, graphs, objects, JSON

- In the relational model, data is represented as tables and fields.

Data Definition Language (***DDL***) allows the user to create data structures in the data model used by the database. A ***schema*** is a description of the structure of the database and is maintained and stored in the ***system catalog***. The schema is ***metadata***.

- A schema contains structures, names, and types of data stored.

Once a database has been created using DDL, the user accesses data using a ***Data Manipulation Language*** (***DML***).

- The DML allows for the insertion, modification, retrieval, and deletion of data.

SQL is a standard DDL and DML for the relational model.

Database Architectures
Not "One Size Fits All"

Relational databases (RDBMS) are still the dominant database architecture and apply to many data management problems.

However, recent research and commercial systems have demonstrated that "one size fits all" is not true. There are better architectures for classes of data management problems:

- Transactional systems: In-memory architectures
- Data warehousing: Column stores, parallel query processing
- Big Data: Massive scale-out with fault tolerance
- "NoSQL": simplified query languages/structures for high performance, consistency relaxation

Database Architectures: NoSQL vs Relational

"NoSQL" databases are useful for several problems not well-suited for relational databases with some typical features:

- **Variable data:** semi-structured, evolving, or has no schema
- **Massive data:** terabytes or petabytes of data from new applications (web analysis, sensors, social graphs)
- **Parallelism:** large data requires architectures to handle massive parallelism, scalability, and reliability
- **Simpler queries:** may not need full SQL expressiveness
- **Relaxed consistency:** more tolerant of errors, delays, or inconsistent results ("eventual consistency")
- **Easier/cheaper:** less initial cost to get started

NoSQL is not really about SQL but instead developing data management architectures designed for scale.

- NoSQL – "Not Only SQL"

Example NoSQL Systems

MapReduce – useful for large scale, fault-tolerant analysis

- Spark, Hadoop, Hive

Key-value stores – ideal for retrieving specific items from a large set of data (architecture like a distributed hash table)

- high-scalability, availability, and performance ; weaker consistency and simpler queries
- Redis, Amazon DynamoDB, Google BigTable, Memcached

Column stores – stores tables with a large number of columns

- Cassandra, HBase

Document stores – similar to key-value stores except store a document (e.g. JSON)

- MongoDB, CouchDB

Graph databases – represent data as graphs

- Neo4J, Amazon Neptune, Microsoft Azure Cosmos DB

Time series databases – optimized for storing time series data often from sensors

- InfluxDB, Kdb+, Prometheus, TimescaleDB

Database Hosting – Where's the Server?

Database hosting selects the machine where the database software executes. Choices:

- Local machine (`localhost`)
- On-premise physical or virtual machine (`cosc304.ok.ubc.ca`)
- Cloud-based hosting (physical/virtual/container) as a service on platforms such as Amazon, Microsoft Azure, Google, Digital Ocean.

Database host must be accessible over the Internet by the clients that connect to it.

Running a Database on `localhost`

Running a database on your machine (`localhost`) is easy. Steps:

- Download and install database software (MySQL, PostgreSQL, etc.)
- Configure and start the database server software

Advantages:

- Full control over database and install process

Disadvantages:

- May not be easy to connect to by clients depending on machine
- Must take time to install/configure database software

Running a Database On-Premise

Running "On-Premise" is when the database is deployed on a (virtual) machine controlled by the organization.

This is often done for security and for performance.

Advantages:

- Data does not leave organization.
- Potential for higher performance.

Disadvantages:

- Organization responsible for deploying, configuring, maintaining, and securing both hardware and database software.

Cloud Databases

Cloud databases are databases hosted by a service provider that allow for easy setup, administration and scaling.

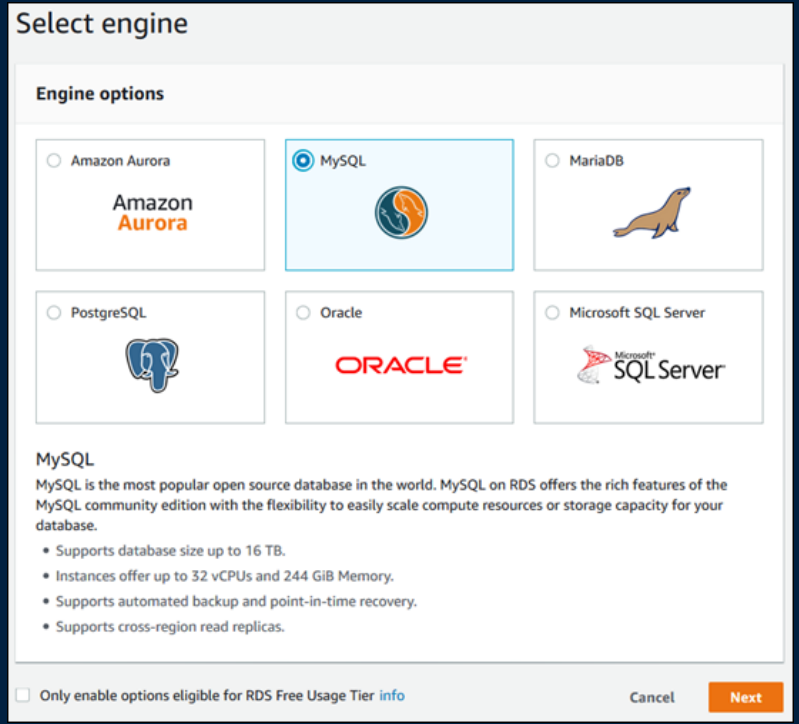
- Database as a service – databases hosted by provider, provide monitoring, backup, fail-over, high-availability, and ability to scale.

Examples: Amazon RDS, Microsoft Azure, Google Cloud, Digital Ocean

Ideal for: Quick start without a server, minimal administration, scaling without expertise

Creating MySQL Instance on Amazon

1) Sign in to AWS Management Console.
   - https://console.aws.amazon.com/rds
2) Select database engine/version.
3) Select database instance size (CPU/memory/storage) and user/password configuration.
4) Configure advanced settings (network accessibility).
5) **Verify price** and Create Database.



Reference:
docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Tutorials.WebServerDB.CreateDBInstance.html

Creating PostgreSQL on Digital Ocean

Sign in to Digital Ocean.

- https://cloud.digitalocean.com

Option #1: User Managed DB

- Create a Droplet.

- Select droplet size and region.

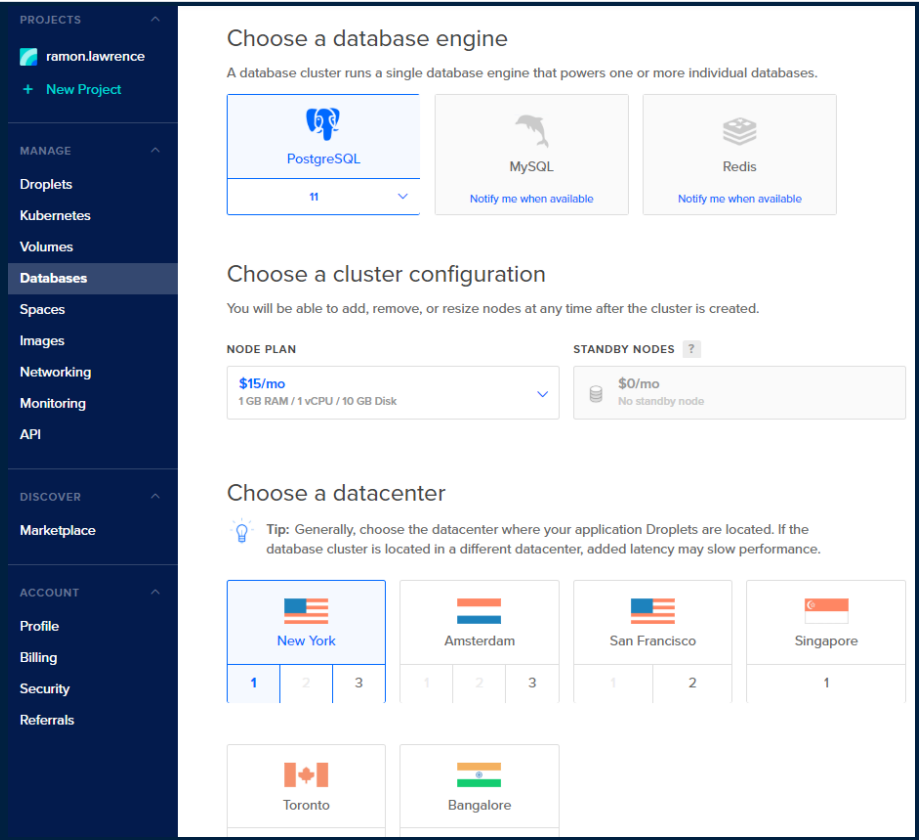
- Login to droplet and install/configure PostgreSQL.

Option #2: Managed DB

- Select Databases and Click Create a Cluster.

- Determine size and region.

- Record login and URL info.



Why this Course is Important

Database systems are a key component of IT infrastructure that are increasingly deployed on cloud providers rather than on premise.

Proficiency in using cloud provider systems and deploying databases is an important skill for employers.

- The course covers a wide variety of databases and cloud providers.

The project and labs will improve your technical communications that are valuable both in industry as well as academia.

Conclusion

A ***database*** is a collection of logically related data managed by a ***database management system*** (DBMS).

NoSQL databases ("Not only SQL") is a category of data management systems that do not use the relational model. There is a variety of NoSQL systems including:

- MapReduce, key-value stores, document stores, graphs, time series, column stores, data lakes

Each database system architecture and implementation has its own benefits and issues. Understanding these features are critical for effective cloud deployments.

Objectives

- Define: database, DBMS, schema, metadata
- Define DDL and DML. What is the difference?
- Understand alternative models for representing data besides the relational model
- List some NoSQL databases and reason about their benefits and issues compared to the relational model for certain problems
- Define database hosting.
- Compare and contrast benefits/challenges with hosting on local machine, on premise, and on a cloud server.


UBC
THE UNIVERSITY OF BRITISH COLUMBIA