Databases, Networks And The Web

Course Notes

Felipe Balbi

July 6, 2020

Contents

Week 1

Key Concepts

- recognise the tools available for this module to edit a node.js file and run it
- describe what static and dynamic web applications are.

1.201 What is a web application?

Data is everywhere around us. For example our account balance is a form of data which gets checked and updated during a commercial transaction.

Web application is a client-server software application in which the user interface runs in a browser[1]. It could be a computer program which allows the user to submit and retrieve data to and from a database.

Static Web Applications

Web applications with little or no interaction with the user.

Dynamic Web Applications

Web applications which allow the user to input, change, and manipulate data.

Desktop vs Web Applications

Desktop	Web
Accessed through OS	Accessed through Web Browser
Different appearance in each OS	Consistent appearenc accross platforms
Fast access to system resources	Slow access to system resources
Lower risk of data loss	Higher risk of data loss
Different version for each OS	Same version accross all platforms
Multiple updates required	Single update for all users

1.203 Further reading

Some useful Further reading is:
`https://techterms.com/definition/web_application`

[1] `https://en.wikipedia.org/wiki/Web_application`

1.205 The life and times of a web request

What happens when we call a web application? We know everything starts with typing the URL in the web browser, but what happens then?

Calling a Web Application

This better shown with the flowchart 1 below.

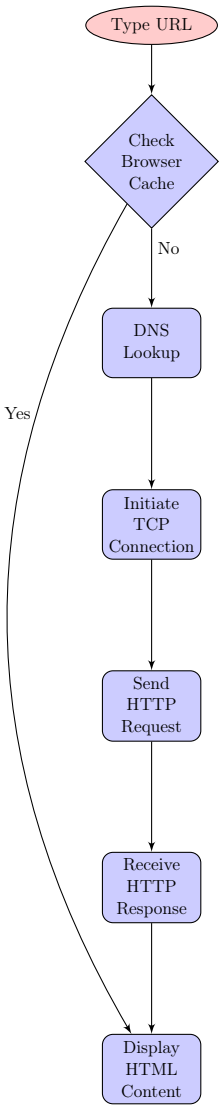

Figure 1: Calling a Web Application

HTTP Protocol

The HTTP Protocol is based on a Request/Response architecture. The client, i.e. the web browser, makes a request for a particular resource. The server returns a response to the browser. This response may or may include the requested resource. This is because errors may occur.

After the browser receives the response, it will evaluate it and decide what and how to display it on the screen.

What if we're dealing with a dynamic web application? Where will the content for the web page come from?

Dynamic web sites are based on programs which run on the web server when an HTTP Request comes in. These programs will generate the content on the fly for the user. One major source of such content is a database.

1.207 Accessing remote resources, HTTP

Ceri, S. et al. Designing data-intensive web applications. (San Francisco, CA: Morgan Kauffman Publishers, 2003). [ISBN 9780080503936].

Chapter 1, p.5-8, Accessing remote resources: the hypertext transfer protocol.

Accessible from here.

Week 2

Key Concepts

- describe what 3-tier web application architecture is.

1.301 Three-tier web application architecture

A three-tier web application architecture is a specialization of the more generic n-tier architecture.

In three-tier architecture, the three modules are as follows:

Presentation Tier Commonly referred to as the *Front-end*, it is responsible for receiving input and displaying output. In general, this part is written in HTML, CSS, JavaScript.

Application Tier Commonly referred to as the *Middleware*, it is responsible for the business logic and calculations. In general, this part is written in JavaScript, PHP, Ruby, Python, and many others.

Data Tier Commonly referred to as the *Back-end*, it is responsible for storing and managing the data the application requires. In general, this part is written in SQL.

The main idea is to keep presentation, application logic and data store separate from each other. This allows each part to be developed and maintained as separate modules.

A web application can be split into two main blocks, depending on where they run:

Client Side The part of the application that runs on the client's computer. Commonly, only the Front-end runs on the client.

Server Side Composed of both the Middleware and the Back-end, it runs on servers (or cloud instances) owned by whoever made the application.

1.304 Information retrieval activity

Ceri, S. et al. Designing data-intensive web applications. (San Francisco, CA: Morgan Kauffman Publishers, 2003). [ISBN 9780080503936].

Chapter 1, section 1.5.7 p.54-55, Three-tier architectures.

Accessible from here.

Week 3

Key Concepts

- explain what a web server is
- use the tools available for this module to edit a simple Node.js web server and run it.

2.001 Introduction to Node.js and Express

During this topic, we create our first web server. The end goal being creating a full web application.

We learn about `Node.js` and `Express.js` which we will use to write our web server.

2.101 Web servers

A web server is a program that uses HTTP to serve web pages in the form of files to their users.

The users send requests to the web servers and its response is a web page.

HTTP, or HyperText Transfer Protocol, is a method for encoding requests and responses. A method of communication between the server and client which defines the rules of interaction.

2.103 Essential reading

Ceri, S. et al. Designing data-intensive web applications. (San Francisco, CA: Morgan Kauffman Publishers, 2003). [ISBN 9780080503936].

Chapter 1, p.5-8, Accessing remote resources: the hypertext transfer protocol

Accessible from here.

2.104 Web server architecture

Figure 2 below presents a common web server architecture. It contains the basic elements of a web server which is composed of the underlying HW, the Operating System, an HTTP Server, a Database, and a Scripting Language Runtime.

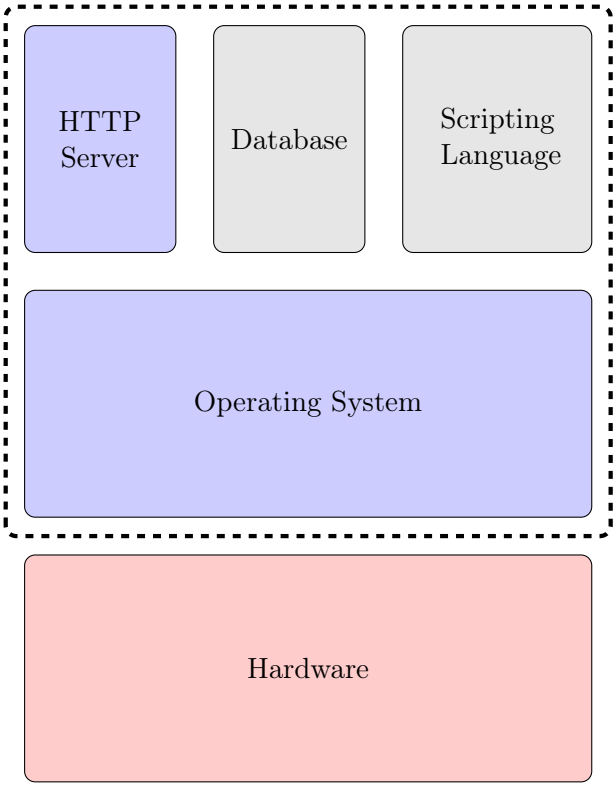

Figure 2: Web Server Architecture

The hardware for a web server could be anything from an embedded, low power device, such as Raspberry Pi or a Beagle Bone, all the way to a set of standard rack-mounted servers running Intel Xeon server-grade CPUs.

The Sofware side of the Web Server includes at least an Operating System, commonly Linux, and an HTTP Server such as Apache, Nginx, etc.

Usually, these also include a Database (MySQL, PostgreSQL, MongoDB, etc) and a Scripting Language runtime (NodeJS, Ruby, Python, Perl, and so on).

2.106 Essential reading

Mendez, M. The missing link: an introduction to web development and programming. (Geneseo, NY: Open SUNY textbooks, 2014). [ISBN 9781502447968].

Refer to Chapter 3 to learn more about web server architecture.

Accessible from here.

2.107 Web hosting

Read about web hosting at the site given below and then answer the questions in the discussion forum.

website.com 'Web hosting: what is web hosting?'

Accessible from here.

2.201 Introduction to Node.js and Express

NodeJS is a server-side JavaScript runtime. It's open-source and cross-platform.

It's implemented as an asynchronous, event-driven runtime environment. Event-driven means that the flow of the program is determined by events received by the program, rather than the sequence of the code. Asynchronous means that NodeJS doesn't wait for responses from external sources.

Because of these characteristics, NodeJS is very good for I/O-intensive applications. It's not, however, a good pick for CPU-intensive applications.

2.204 Node.js

Krause, J. Programming web applications with Node, Express and Pug. (New York, NY: Apress Media LLC, 2017). [ISBN 9781484225110]

Chapter 3 Introduction to Node, pp.15–46. You can refer to this chapter throughout Topics 2–4.

Accessible from here.

Open.js foundation 'AboutNode.js'.

Accessible from here.

Week 4

Key Concepts

- use the tools available for this module to edit an Express.js web server and run it.

2.503 Express

Yaapa, H. Express web application development: learn how to develop web applications with the Express framework from scratch. (Birmingham: Packt Publishing Ltd., 2013). [ISBN 9781849696555]

- Chapter 1, pp.24–26, What is Express?
- Chapter 1, pp.30–36, The stuff that makes up Express
- Chapter 2, pp.51–55, Your first Express app

Accessible from here.

2.601 Summary and Further reading

Krause, J. Programming web applications with Node, Express and Pug. (New York, NY: Apress Media LLC, 2017). [ISBN 9781484225110]
Accessible from here.

Week 5

Key Concepts

- describe and apply routing in web server development
- describe and apply the 'separation of concerns' programming principle in your web application development.

3.201 Separation of concerns (SoC)

SoC is the idea that each module in a web application should be responsible for one thing.

It's a design principle where we aim at isolating modules from each other and have each module address a separate concern.

A concern can be "presenting data to the user" or "connecting to a database".

In order to implement a web application using this design principle, we must first define the concerns. Then we can design a separate module or layer for dealing with that concern. One thing to remember is that we also aim at minimizing coupling of the different concerns by making a minimal interface through which the modules communicate.

SoC is about planning and designing logical layers in a way that it is loosely coupled by interfaces with other layers. What we mean by that is that modules should indepent from each other, thus allowing any module to be replaced by another module that implements the same interface.

SoC can be expressed at different levels: functions, modules, controls, widgets, tiers, services, etc. It's not limited to large layers within an application.

SoC helps us reduces complexity by means of encapsulation, it aids us with keeping up with the DRY (*Don't Repeat Yourself*) principles, improves portability of the application and improves maintainability and testability of the application.

Week 6

Key Concepts

- describe and apply templating in web application development.

3.401 Templating engines

Templating engines allow us to produce content dynamically while also allowing us to keep HTML separate from JavaScript.

With that, we can produce Web Applications that are more interactive and respond to users' input.

In summary, our web application will contain static template files containing static content and variables. During runtime, the template engine will replace the variables with actual values and compile HTML files to be served to the user.

Usually, template engines also allow us to render e.g. a list of elements from a single template by using a loop.

Some popular template engines are:

- EJS
- Pug
- Mustache

During this course we focus on EJS.

An EJS file is just a regular HTML file renamed to have the extension `.ejs`. It contains two special tag elements, they are:

<%= %> Used to output the return value of a javascript expression into the document

<% %> Used to evaluate an expression but it won't add the result of evaluation to the document.

3.407 Further reading, Pug

For more information on the Pug templating engine refer to the following:

Krause, J. Programming web applications with Node, Express and Pug. (New York, NY: Apress Media LLC, 2017). [ISBN 9781484225110].

- Chapter 6, Introduction to Pug
- Example pug templates pp.83, 84, 85
- Chapter 7, Language components of Pug

Accessible from here.

3.501 Summary and Further reading

In this topic you learned about:

- the separation of concerns (SoC) principle of programming
- organising your web applications into separate folders
- templating engines.

Some Further reading you may be interested in is:
Krause, J. Programming web applications with Node, Express and Pug. (New York, NY: Apress Media LLC, 2017). [ISBN 9781484225110].

- Chapters 3, 4 and 5

  Accessible from here.

Yaapa, H. Express web application development. (Birmingham: Packt Publishing Ltd., 2013). [ISBN 9781849696555].

- Chapters 1, 2 and 3

  Accessible from here.

Week 7

Key Concepts

- describe and apply form handling in your web application.

4.001 Introduction

Input forms are one way to provide user interaction in a dynamic web application. We learn how to create a form using HTML tags and how to display it to the users in the front end of a dynamic web application.

4.103 Form-data validation

Read about validation in JavaScript:

- `https://www.tutorialspoint.com/javascript/javascript_form_validations.htm`

Read about express-validator module:

- `https://www.npmjs.com/package/express-validator`

Try to find answers to these questions:

- What is form-data validation?
- What are the different types of form data validation?
- How can it be achived?
- How can express-validator module facilitate form-data validation?

Week 8

Key Concepts

- describe and apply GET and POST request methods in your web application
- describe and run the code to retrieve form data in middleware.

4.201 GET and POST request methods

GET and *POST* are two of the main HTTP methods. *GET* is used to request data from a resource on a web server while *POST* is used to send data to create or update a resource on a web server.

More information about HTTP methods can be found here.

4.203 HTTP methods

Yaapa, H. Express web application development. (Birmingham: Packt Publishing Ltd., 2013). [ISBN 9781849696555].

pp.86–87, A quick introduction to HTTP verbs

Accessible from here.

4.301 Summary and Further reading

In this topic you learned about:

- creating forms using HTML
- accessing forms from middleware (res.send(), res.render())
- collecting form data in middleware (main.js) in Node.js
- collecting form data when the request method is GET
- collecting form data when the request method is GET.

Some Further reading you may be interested in is:

Krause, J. Programming web applications with Node, Express and Pug. (New York, NY: Apress Media LLC, 2017). [ISBN 9781484225110].

Accessible from here.

Week 9

Key Concepts

- describe what a list of data is
- describe modification problems related to lists of data.

5.001 Introduction

We start studying databases. We will learn the basics of database systems, SQL and MySQL.

5.101 Introduction to databases

Databases have **storage** as the main purpose, however data retrieval, i.e. searching, is another important feature to take into consideration.

We can also access the database to create, modify or delete data. Sometimes the acronym *CRUD* standing for Create, Read, Update, and Delete, is used ot remember the main operations on a database.

Another purpose of a database is to provide structure for our data. Data that's well-structured is easier to be queried.

Before moving forward, let's define *Database* and *DBMS*:

Database A collection of data

DBMS A DataBase Management System is software which allows us to manage and interact with the data.

Not every collection of data is referred to as a database. In general a database is a collection of data that "is highly valuable, relatively large and accessed by multiple users at the same time".

With that in mind, the role of a DBMS is to provide a way to store and retrieve information that's convenient and efficient.

5.103 Data and databases

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

- Chapter 1, pp.38–43, Data matter!
- Chapter 1, pp.60–63, Evolution of database systems
- Chapter 1, pp.63–67, The range of database applications

Accessible from here.

Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- Chapter 1, pp.33–39, Database and database users

Accessible from here.

5.104 Lists of data

Before the creation of databases, data was stored in linear files. This consisted of long lists of tab delimited values or lists.

List-based storage systems don't scale well as the amount of data to be stored grows. One of the issues is that of *Data Redundancy*. Another issue arises when we want to delete an entry from a list.

The small table below provides a short summary of issues with lists:

Problem	Example
Insertion Problems	Data Redundancy
Deletion Problems	Information Loss
Update Problems	More than one field must be updated

Week 10

Key Concepts

- describe what a relational database is.

5.201 Introduction to relational databases

As mentioned before, before the introduction of relational databases, data was kept in lists which made data retrieval a very laborious task.

Relational databases, therefore, were created to make the process simpler and quicker.

A relational database is a very simple concept. In a relational database data is organized in tables which relate (or refer to) each other.

Relationships between these tables are represented by common values in related tables.

Figure 3 depicts a relational database model. We can see that each object is represented by a table. In figure 3 we have two objects: Article and Author. Each column in a table is called a *field* and each row is called a *record*.

Author Table		
id	firstname	lastname
0	Sheela	Begum
1	Nyambi	Walubita

Article Table		
id	Title	Author
0	Databases	1
1	OOP	0

Figure 3: Relational Database

As we can see, all the author data is kept in a single table, so the data redundancy problems are solved. Moreover, the insertion, update, and deletion problems are also addressed. Note that we make the relation between tables by using the unique id field.

Databases are queried using *SQL* or Structured Query Language. This language was specifically designed for the purpose of retrieving data from a DBMS.

5.306 SQL

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

- Chapter 6, pp.279–295, Introduction to SQL

Accessible from here.

Week 11

Key Concepts

- Describe and apply statements in SQL to do basic database oprations.

6.001 Introduction

In this topic, all we do is work with the `MySQL` shell. We will update and delete records in the database and discuss the concept of *Primary Keys*. We also learn how to access the database from Node.js code.

6.103 CREATE, SELECT and INSERT in SQL

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

- Chapter 6, pp.287–88, Defining a database in SQL
- Chapter 6, pp.289-290, Creating tables

Accessible from here.
Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- Chapter 5, section 5.3.1, pp.196–97, The insert operation
- Chapter 6, section 6.3, pp.217–21, Basic retrieval queries in SQL

Accessible from here.

6.207 Basic database operations in SQL

Refer to the links below for more information on CREATE DATABASE, CREATE TABLE and SELECT in SQL:

- `https://www.w3schools.com/sql/sql_create_db.asp`
- `https://www.w3schools.com/sql/sql_create_table.asp`

- `https://www.w3schools.com/sql/sql_select.asp`

Refer to the following links for more information on update and delete operations in SQL:

- `https://www.w3schools.com/sql/sql_ref_update.asp`
- `https://www.w3schools.com/sql/sql_delete.asp`

In addition to the above, refer to:
Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- Chapter 6, Section 6.3, pp.218–28 Basic retrieval queries in SQL
- Chapter 6, Section 6.4, pp.228–31 INSERT, DELETE and UPDATE statements in SQL

Accessible from here.

Week 12

Key Concepts

- Describe and apply primary key and foreign keys in relational databases
- Recognise and apply access to databases from middleware code in Node.js to build dynamic web applications

6.301 PK (primary keys)

In the context of Relational Databases, a Primary Key is a non-empty set of fields that uniquely identify a row.

SQL has a way of defining primary keys during table creation. In the example below, *id* is the primary key in table *books*.

```
1 create table books (id int auto_increment,
2                     name varchar(50),
3                     price decimal(5, 2) unsigned,
4                     primary key(id));
```

Whenever we're designing a new database table, thought must be given to the choice of primary keys. It's very common to introduce an auto-incrementing integer field called *id* and that should be our first choice.

There are some rules imposed by the DBMS, commonly referred to as *Integrity Constraints*. They can be used to limit the range of acceptable values (e.g. *not null* constraint).

Some integrity constraints are related to the Primary Keys and are, therefore, called Primary Key Constraints. Such constraints say that every table must have a primary key (or set of primary keys).

6.411 Sanitisation

Refer to the link below and read about sanitisation or sanitisation in form handling in Node.js using the Express-sanitizer module:

`https://www.npmjs.com/package/express-sanitizer`

6.501 Summary and Further reading

Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- If you are interested in the subject of database security, you can refer to Chapter 30.
- If you are specifically interested to know more about 'SQL injection' as one of the threats against database security, please refer to section 30.4, pp.1172–76.

Accessible from here.

Week 13

Key Concepts

- Describe and apply passing variables from middleware to backend (database)
- Describe and apply passing variables from middleware to front-end (templates)

7.001 Introduction

Dedicated to Midterms assignment.

7.103 Database design

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

- Chapter 1, pp.67–77, Developing a database application for Pine Valley furniture company

Accessible from here.

7.201 Summary and Further reading

Ceri, S. et al. Designing data-intensive web applications. (San Francisco, CA: Morgan Kauffman Publishers, 2003). [ISBN 9780080503936].

- Chapter 6, pp.193–200, Overview of the development process.

Accessible from here.

Week 15

Key Concepts

- Explain and apply inserting records in junction tables
- Explain the context behind the relational model and describe the process of schema development
- Explain the context behind the junction (bridge) tables in relational model and describe the process of schema development

8.101 Entity relationship diagrams and junction tables

To design databases, we can make use of *Entity Relationship Diagrams* and *Junction Tables*.

When designing a database, we must decide which tables are the building blocks of the relational database. Usually, we have a set of requirements to start. We can, then, divide the information into entities, then we design a table for each these entities by considering which information should be stored in them. From there, we make a decision on primary keys. Each and every table should have a primary key.

Next, we design the relationships between the tables by adding Foreign Keys.

An Entity Relationship Diagram, or ER Diagram for short, can help us designing databases. More information about ER Diagrams can be found here.

There are four types of relationship cardinalities:

- Optional-one
- Optional-many
- Mandatory-one
- Mandatory-many

Many-to-many associations create a problem: how will we know how many foreign keys we should allocate in a table? A solution to this problem is a Junction Table. It's essentially an extra table which simply maps between two other tables. That way, in a restaurant system, a customer can place many orders each containing many food items.

8.105 Entity relationship (ER) diagrams

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

- Chapter 2, pp.89–92, Introduction
- Chapter 2, pp.92–95, The E-R model: an overwiew

Accessible from here.

8.201 Foreign keys and SQL join

Remember a Primary Key provides a unique identify for each record in a table. Foreign Keys are used by other tables to refer to these records.

To add a Foreign Key with MySQL we would use a command like below:

```
create table book (id int auto_increment,
                   name varchar(50),
                   price decimal(5, 2), unsigned,
                   int publisher_id,
                   primary key(id),
                   foreign key(publisher_id) references publisher(id));
```

When selecting data, we can use the foreign key to fetch the publisher data when fetching book data using a *SQL JOIN*. The example below gives an idea of how to construct such queries.

```
select book.name, book.price, publisher.publisher_name
  from book join publisher
    on book.publisher_id = publisher.id
 where book.price >= 30
```

8.205 Foreign keys and SQL JOIN

Hoffer, J., R. Venkataraman and H. Topi Modern database management. (Harlow: Pearson Education Ltd., 2016). 12th Global edition. [ISBN 9781292101866].

Chapter 4, pp.192–200, Logical database design and the relational model.

More specifically:

- p.194 introduces foreign keys
- p.196, Integrity constraints
- p.198, Referential integrity
- p.199, Creating relational tables.

Accessible from here.

Week 16

Key Concepts

- Explain and apply inserting records in junction tables

8.301 Summary and Further reading

In this topic you have learn about:

- Entity relationship (ER) diagram in relational databases
- Association types between entities in an ER diagram
- The problem with many-to-many association
- Junction or bridge tables in ER diagrams and how they can solve the problem with many-to-many associations
- Foreign keys in relational databases
- Querying multiple table in relational databases by using SQL JOIN statement

Further reading:
Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- Chapter 3, pp.89–135.

Accessible from here.

Week 17

Key Concepts

- Perform queries on databases using multiple related tables by JOIN queries
- Perform advanced queries on databases using aggregate functions

9.001 Introduction

SQL has aggregate functions, some of which are *MIN*, *MAX*, *SUM*, and *COUNT*. These queries work on a set of records and return a single number. They are useful for generating records or performing simple calculations that are needed often.

We will also learn about two new types of *JOIN* statements in SQL: *LEFT JOIN* and *RIGHT JOIN*. Furthermore, we will look at nested *SELECT* queries.

9.105 Aggregate functions

Elmasri, R. and S. Navathe Fundamentals of database systems. (Harlow: Pearson Education Ltd., 2017). 7th Global edition. [ISBN 9781292097619].

- Chapter 7, Section 7.1.7, pp.246–248, Aggregate functions in SQL

Accessible from here.

9.201 SQL left and right joins

To make the differences between the different SQL *JOIN* statements, a pictorial representation follows in figures 4, 5, and 6.

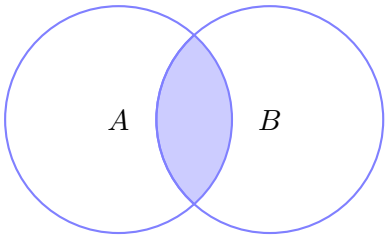

Figure 4: *SELECT * from A JOIN B ON A.B_id = B.id*

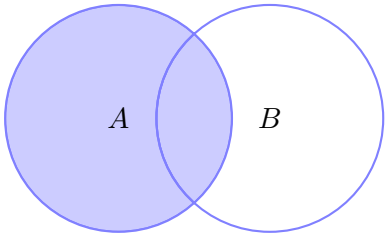

Figure 5: *SELECT * from A LEFT JOIN B ON A.B_id = B.id*

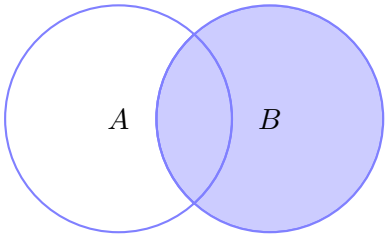

Figure 6: *SELECT * from A RIGHT JOIN B ON A.B_id = B.id*