SQL Notes

Select Statement

* Used when asking for data
* select \* from player
* \* is the entire database
  + Can be replaced with fields wanted separated by commas (selects certain columns)
  + select player\_name, birthday from player
* distinct keyword gets only unique results
  + select distinct player\_name from player

As Statement

* select player\_name as name from player
* then player\_name will be named just “name”

Where Clause

* Used to select certain rows from the database
* select \* from player where weight = 190
  + only returns rows (players) with a weight of 190
  + can also use comparison operators ><
  + also and + or for multiple scenarios
* select \* from player where player\_name like ‘Aaron%’
  + gets all player\_name fields that start with Aaron
  + %Aaron gets ending and Aaron% gets starting with, %Aaron% gets any containing, A%n gets all starting with A and ending with n
* Between operator for integers
* In operator for strings
* Is null operator for null items
  + Also is not null operator

Order By

* Sort the data by a certain things
* Select \* from player order by weight desc (descending)
  + Default is ASC (ascending)

Aliasing Tables

* Enter allias after the actual name and then you can use the allias everytime the actual would typically be called

Group By Clause

* Takes all records, and groups them by a single column
* SELECT column FROM table GROUP BY columnToGroupBy

Having Clause

* Just like the Where clause but it goes after the group by clause

RDMS

* Relational Database Management System

Primary Key - the minimum number of columns to uniquely identify a record (row), identifies an individual record in a table

Creating/Editing/Deleting Database information

* Create a database
  + CREATE DATABASE databaseName
* Delete a database
  + DROP DATABASE databaseName
* Create a table in your database
  + CREATE TABLE tableName(columnName columnType)
  + Use commas to separate if there are multiple columns
    - Add NOT NULL if it needs to have a value
    - Always a good idea to include an ID
      * CREATE TABLE tableName(id INT NOT NULL IDENTITY(1,1))
      * Add PRIMARY KEY (id) to set it as the primary key
      * Add FOREIGN KEY (idname) REFERENCES tableName (id)
* Delete a table
  + DROP TABLE tableName
* Insert data
  + INSERT INTO tableName (columnName) VALUES (value1),(value2)
* Update date
  + UPDATE tableName SET columnName = newValue WHERE (select what record(s) you want to change)

Joining Tables

* SELECT \* FROM leftTable [TYPE] JOIN rightTable ON rightTable.column = leftTable.column
* Inner Join
  + Combines data that has both a value on the left and right tables, and only returns data that match between tables
  + Is the default join type
* Left Join
  + Returns everything from the left table even if they don’t have a match on the right table
* Right Join
  + Returns everything from the right table even if they don’t have a match on the left table
* Full Join
  + Returns everything from both tables, whether or not anything has a match

Aggregate Functions

* Sum
  + SELECT SUM(column) FROM table
  + Adds all entries from the selected column
* AVG
  + SELECT AVG(column) FROM table
  + Averages all entries from the selected column
* COUNT
  + Useful with Group By
  + SELECT column1, Count(column2) FROM table GROUP BY column1
  + The count happens on each group instead of the whole table