Project 2

# Project Info

## Due Date

End of Week 5 : Saturday Oct. 6th

**LATE SUBMISSIONS WILL NOT BE ACCEPTED**Submission: Submit to the Project 2 drop box.

See “Submission Requirements” below for details on how to generate the required file.

**This is an individual assignment. Significant overlap between submissions from two or more students may be flagged for plagiarism.**

## Grading

|  |  |  |
| --- | --- | --- |
| **Item** | **Description** | **Weight** |
| FK Indexing | Indexes across foreign keys, including junction tables, are correct. | 35% |
| Query Indexing | Indexes for identified queries are correctly created. | 15% |
| Unique Constraints | Alternate keys and unique indexes exist for all required columns. | 25% |
| Check Constraints | All required check constraints exist and are appropriate. | 15% |
| Defaults | All required defaults exist and are appropriate. | 10% |
| Redundant Indexes | Penalty for overlapping indexes | Up to -20% |
| Error on Execution | Penalty for failure to run without modification | Up to -20% |
| Submission Requirements | Penalty for failure to adhere to the submission requirements | Up to -10% |
| **Total** |  | **100%** |

## Business Requirements

You have been provided with a partially completed database for a system that tracks a bike sharing program. After some testing, it has been determined that the structure alone is insufficient to properly constrain the database. Testing also revealed performance concerns. You have been asked to add constraints and indexes to satisfy the following business and technical requirements.

The customer has told you that whenever bikes are checked in or out, a scanner reads the **serial number** which is used to **uniquely identify** that bike in the database. Lookups on SerialNumber will need to be properly optimized and constrained.

In testing, the customer found that a **station's capacity** would sometimes incorrectly be set to a negative number. Since this is invalid, they would like to prevent negative numbers from being added to the Capacity column. Testing also found that station records were frequently looked up by StationName, which can be used to **uniquely** identify records. They would like these lookups optimized.

The customer has identified that dates in the system (Riders.DateOfBirth and Accounts.AccountOpenDate) should never be a future date, so they must be equal to or less than the current datetime. When testing these tables the customer discovered that Accounts will often be uniquely identified by AccountNumber, and Riders may be uniquely identified by RiderSIN. These lookups will need to be optimized.

To improve consistency, the customer has asked you to provide some suitable **defaults** when complete account details are not available. When not provided, CurrentBalance should be set to 0 and AccountOpenDate should be set to the current date.

A review of queries on the Addresses table identified three queries that need to be optimized. The first query creates a sorted list of provinces and cities. This list is **sorted by province first, then by city**. The second query looks up records by province only. The last query looks up records by city only.

\*\*COME BACK TO\*\*

The customer has found that lookups will frequently be done in **both directions** across all foreign keys. They would like lookups by the parent and child columns optimized for all foreign keys. (A ****parent**** is the table that stores the primary key, A****child****is any table that references the parent with a foreign key.) They would also like you to optimize lookups in **both directions** across the two junction tables. Covering indexes should be provided to quickly look up Riders by Addresses, Addresses by Riders, Riders by Accounts, or Accounts by Riders.

## Technical Requirements

In addition to satisfying the business requirements, you have been asked to follow these technical standards.

1. You must work with the schema provided in the Project 2 folder. You may not use your own.
2. You should not be altering the existing schema, including the database name. The correct solution will only add new constraints and indexes.
3. **There should be no duplication of indexes. If an index is entirely covered by another index it should be removed. Remember, some indexes are automatically generated by SQL Server.**
4. **Indexes may not include any redundant columns.**
5. The function **GETDATE**() should be used to get the current datetime.
6. You will submit the entire database using the instructions below. This includes the database that was provided to you along with the modifications you made to satisfy these requirements.

# Submission Requirements

You will submit a database script generated by SQL Server. You do not need to submit the code you used to create your database, **but you are permitted to do so if you would like**.

**Make sure you are satisfied with the database you have created before scripting it out.**

Script out the database:

1. Right click on your database and select “Tasks->Generate Scripts…”
2. If you see an introduction page, click “Next >”
3. On the “Choose Objects” page, select **“Script entire database and all database objects”.**
4. Click “Next >”
5. On the “Set Scripting Options” page, ensure that “Save scripts to a specific location”, “Save to file”, and “Single file” are all selected.
6. Change the filename so that it is your first and last name with a .sql extension (e.g. “BruceWayne.sql”)
7. Click “Next >”
8. Review your selections and click “Next >”

This will create a .sql file. Submit this file to the dropbox.

**You must test your script before submitting. (ie. Re-run it in SQL Server Management Studio to ensure it executes).**