ENFORCING DATA INTEGRITY

Enforcing Data Integrity

- A database is only as valuable as the accuracy of information it contains
- Goal: Improve data accuracy by preventing invalid data entry

Choose Appropriate Data Types

- Use numeric types for truly numeric data (data that represents a quantity or will be used in numeric calculations)
- Do not use numeric types for quasi-numeric data (Zip codes, Social Security Numbers)
- Always use DATE or DATETIME type for dates -- never text or numeric type
- Use integer type for sequentially generated primary keys. Consider allowing the database to auto-generate these primary key values.

Enforce Uniqueness

- Databases automatically enforce uniqueness on primary key columns
- Use a UNIQUE constraint in the CREATE TABLE statement to make the database enforce uniqueness on candidate key columns
- Example:

```
CREATE TABLE Person (
   Person_ID                INTEGER  PRIMARY KEY,
   Social_Security_Num CHAR(11) UNIQUE,
   First_Name                VARCHAR(20),
   Last_Name                 VARCHAR(20),
   UNIQUE(First_Name, Last_Name)
)
```

Default Values

- MySQL can auto-generate primary key values when you define a primary key column as an AUTO_INCREMENT column.
- You can define a default value for a column, so that if the user does not provide one when inserting a record, the database automatically supplies it.
- This is especially useful for date columns:
  - CREATE TABLE MyTable (

    ...

    creation_time DATETIME DEFAULT CURRENT_TIMESTAMP

)

Required Values

- Mark columns NOT NULL if they should contain a value
- Note: VARCHAR columns that are NOT NULL may contain zero-length strings

Foreign Keys

- Defining foreign keys is an important data integrity tool
- Define code tables and reference them with foreign keys to require that columns containing codes must contain a valid code