**SMG – Essential Database Naming Conventions**

**Style**

1. use lowercase characters
   * eliminates question of proper case as well as errors related to case-sensitivity
   * speeds typing rate and accuracy
   * differentiates table and field names from uppercase SQL keywords
2. separate words and prefixes with underlines, never use spaces
   * promotes readability (e.g. book\_name vs. bookname)
   * avoid having to bracket names (e.g. [book name] or `book name`)
   * offers greater platform independence
3. avoid using numbers
   * may be a sign of poor normalization, hinting at the need for a many-to-many relationship

**Table Names**

1. choose short, unambiguous names, using no more than one or two words
   * distinguish tables easily
   * facilitates the naming of unique field names as well as lookup and linking tables
2. give tables singular names, never plural (**update:** i still agree with the reasons given for this convention, but most people really like plural table names, so i’ve softened my stance)
   * promotes consistency with naming of primary key fields and lookup tables
   * ensures alphabetical ordering of a table before its lookup or linking tables
   * avoid confusion of english pluralization rules to make database programming easier (e.g. activity becomes activities, box becomes boxes, person becomes people, data remains data, etc.)
   * more grammatical SQL (e.g. SELECT activity.activity\_name –rather than– SELECT activities.activity\_name)
3. avoid abbreviated, concatenated, or acronym-based names
   * promotes self-documenting design
   * easier for developer and non-developer to read and understand
4. prefix lookup tables with the name of the table they relate to
   * groups related tables together (e.g. activity\_status, activity\_type, etc.)
   * prevents naming conflicts between generic lookup tables for different entities
5. for a linking (or junction) table, concatenate the names of the two tables being linked in alphabetical order
   * orders linking table with a related entity table
   * expresses composite purpose of the table
   * this should be waived if the linking table has a natural, standard, or obvious name (e.g. “item” in: [order] 1 to M [item] M to 1 [product])
   * this must be waived if there are multiple linking tables between the same two tables, (e.g. “student” and “instructor” between the tables “person” and “class”)
6. rare problem
   * what about a linking table for a generic lookup table?
     + this can be avoided by renaming the generic lookup (and removing the prefix)
     + suggestion: as a variation on the linking table naming convention above, use a special character (such as a plus-sign “+”, a hyphen “-”, or a double-underline “\_\_”) to separate concatenated table names (e.g. the linking table for activity and activity\_type would be activity\_\_activity\_type)

**field/column names**

1. the primary key should be the singular form of the table name suffixed with “\_id” (**update:** i now just name all my auto-increment primary keys “id”)
   * allows primary key to be deduced/recalled from the table name alone (e.g. primary key of the product table would be product\_id)
   * consistent with the name of the foreign key
   * prevents having to alias primary keys in programming
2. prefix the name of every field with the table name, excluding foreign keys (**update:** i no longer follow this convention because it’s very tedious to type, and because it’s obviated by the use of abbreviated table aliases in SQL)
   * prevents using “name”, “order”, “percent”, etc. as field names and clashing with SQL/RDBMS reserved words
   * creates near unique field names (e.g. product\_name, product\_code, product\_description, etc., often simplifying query design and SQL coding, recommended for PHP)
   * makes the field names consistent with the primary key
   * differentiates foreign key fields from fields native to the table
   * maintains semantic transparency of field names when using table aliases (e.g. SELECT a.activity\_name FROM activity a)
   * prevents naming a field the same name as the table
   * this can be waived for databases with many tables (30+), tables with many fields (30+), and long and obviously unique field names
   * this can be waived if your database programming always refers to fields in the form *tablename.fieldname*
3. foreign key fields should have the same name as the primary key to which they refer (**update:** obviously now that i just name my primary keys “id”, i name my foreign keys the singular form of the table name + “\_id”)
   * makes the table to which they refer completely obvious
   * if there are multiple foreign keys referencing same table, prefix the foreign key field name with an appropriately descriptive adjective (e.g. lead\_person\_id, technical\_person\_id, etc. which transparently reference person\_id in the person table)
4. suffix fields of type date with “\_on”, suffix fields of time datetime with “\_at”, and prefix fields of type boolean with “is\_” or “has\_”
   * prevents confusing with more common text/number data types