データベース設計ルール

テーブル：

1. 全部小文字
2. 略称しない
3. 単語と単語の間に「　\_　」で分ける

コメント：他のケースもある。例えば：PascalCasing。

1. 多対多の中間表の名前は関連の２つの表の組み合わせ

　　例：student\_course

1. 子システムのテーブルは前に修飾略称をつける

　　例：審査（audit）子システム、aud\_

1. 単数（singular）形
2. 関連表の名前は２つの主表の名前の組み合わせである。

例；table1: student table2: teacher connection: student\_teacher

コラム：

１．単語と単語の間に「　\_　」で分ける

コメント：他のケースもある。例えば：PascalCasing。  
２．略称しない

主キー：

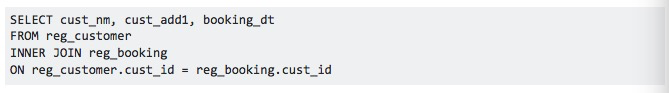
１．テーブル名\_id

　　例：学生表の主キー、student\_id

注意：主キーも単数！

外部キー：

１．外部キーのフィルドは主キーのフィルドと同じである

　　例：

外部キー制限（foreign key constrains）：

１．fk\_ParentTable\_ChildTable

　　例：fk\_teacher\_student

注意：外部キーも単数！

インデックス

１．Idx\_

コーディングルール

１．パッケージの名前：

コントローラーレイヤー：イベント処理などを扱うクラス（リクエスト処理）

src/com/sic/ocms/action/\*

サービスレイヤー:業務ロジック

src/com/sic/ocms/service/\*

DAOレイヤー：data access object(データの追加、アップデートなどの操作のクラス)

src/com/sic/ocms/dao/\*

persistenceレイヤー：エンティティクラス

src/com/sic/ocms/persistence/\*

クラスの名前：

　名前の付け方

　データベースのテーブル名と一緒

　パスカルケースを使う

　例：PascalCasing

属性の名前：

　データベースのカラム名と一緒

　キャメルケースを使う

　例：CamelCasing

getとset メソット：

まず、getを生成して、setを生成する。

￼￼