The following list provides additional information about other `SELECT` clauses:

- A *`select_expr`* can be given an alias using `AS` *`alias_name`*. The alias is used as the expression's column name and can be used in `GROUP BY`, `ORDER BY`, or `HAVING` clauses. For example:

  ```
  SELECT CONCAT(last_name,', ',first_name) AS full_name
    FROM mytable ORDER BY full_name;
  ```

  The `AS` keyword is optional when aliasing a *`select_expr`* with an identifier. The preceding example could have been written like this:

  ```
  SELECT CONCAT(last_name,', ',first_name) full_name
    FROM mytable ORDER BY full_name;
  ```

  However, because the `AS` is optional, a subtle problem can occur if you forget the comma between two *`select_expr`* expressions: MySQL interprets the second as an alias name. For example, in the following statement, `columnb` is treated as an alias name:

  ```
  SELECT columna columnb FROM mytable;
  ```

  For this reason, it is good practice to be in the habit of using `AS` explicitly when specifying column aliases.

  It is not permissible to refer to a column alias in a `WHERE` clause, because the column value might not yet be determined when the `WHERE` clause is executed. See Section B.3.4.4, “Problems with Column Aliases”.

- The `FROM` *`table_references`* clause indicates the table or tables from which to retrieve rows. If you name more than one table, you are performing a join. For information on join syntax, see Section 13.2.10.2, “JOIN Clause”. For each table specified, you can optionally specify an alias.

  ```
  tbl_name [[AS] alias] [index_hint]
  ```

  The use of index hints provides the optimizer with information about how to choose indexes during query processing. For a description of the syntax for specifying these hints, see Section 8.9.4, “Index Hints”.

  You can use `SET max_seeks_for_key=`*`value`* as an alternative way to force MySQL to prefer key scans instead of table scans. See Section 5.1.8, “Server System Variables”.

- You can refer to a table within the default database as *`tbl_name`*, or as *`db_name`*.*`tbl_name`* to specify a database explicitly. You can refer to a column as *`col_name`*, *`tbl_name`*.*`col_name`*, or *`db_name`*.*`tbl_name`*.*`col_name`*. You need not specify a *`tbl_name`* or *`db_name`*.*`tbl_name`* prefix for a column reference unless the reference would be ambiguous. See Section 9.2.2, “Identifier Qualifiers”, for examples of ambiguity that require the more explicit column reference forms.

- A table reference can be aliased using *`tbl_name`* `AS` *`alias_name`* or *`tbl_name alias_name`*. These statements are equivalent:

  ```
  SELECT t1.name, t2.salary FROM employee AS t1, info AS t2
    WHERE t1.name = t2.name;

  SELECT t1.name, t2.salary FROM employee t1, info t2
    WHERE t1.name = t2.name;
  ```

- Columns selected for output can be referred to in `ORDER BY` and `GROUP BY` clauses using column names, column aliases, or column positions. Column positions are integers and begin with 1:

  ```
  SELECT college, region, seed FROM tournament
    ORDER BY region, seed;
  ```