Schema for examples:

```
apply(sid, cname, major, decision)
college(cname, state, enrollment)
student(sid, sname, gpa, sizehs)
```

1. **Filter columns** The select clause controls which attributes are included in the query answer. Let's query for student names and their GPAs.

   ```
   select sname, gpa from student;
   ```

   A * is short hand for all columns.

   ```
   select * from student;
   ```

2. **Derive new columns** You can also use the select clause perform mathematical (and other) operations on the attribute values. Let's make a new attribute called *scaled GPA*. Idea is to increase GPA for those at large high schools, specifically scaledGPA = (GPA × sizeHS/1000). Use the `AS` keyword to name the attribute `scaledgpa`.

   ```
   select sname, (gpa * sizehs / 1000) as scaledgpa
   from student;
   ```

3. **Eliminate duplicate rows** Use the distinct keyword to filter duplicate rows.

   ```
   select distinct sid, cname from apply where sid < 500;
   ```

4. **Filter rows** Use the where clause to filter the results. Find applications where the major is CS and the college is either Colgate or Bucknell. Use parentheses to control operator precedence.

   ```
   select * from apply
   where major = 'CS' and (cname = 'Colgate' or cname = 'Bucknell');
   ```

   When filtering on a string attribute, you can also search for certain patterns rather than exact string literals. Use the `LIKE` keyword and the % character. The % character is a "wildcard" that matches any sequence of zero or more characters. Find applications to colleges whose name starts with 'C'.

   ```
   select *
   from apply
   where cname LIKE 'C%';
   ```

5. **Limit results** Use the limit clause to control the number of rows returned.

   ```
   select * from apply
   where major = 'CS'
   limit 3;
   ```

6. **Ordering rows** Use the order by clause to control the order of rows returned. Find accepted applications ordered by college name.

   ```
   select *
   ```

```
from Apply
where decision = 'Y'
order by cName;
```

7. **Groups and aggregations** Use the group by clause to group together rows with identical values in specified columns. Use *aggregation functions* to summarize attribute values of group members. Aggregation functions include: COUNT, MIN, MAX, SUM, AVG. Find number of accepted applications to each college.

   ```
   select cName, count(*) as numAccepted
   from Apply
   where decision = 'Y'
   group by cName;
   ```

   What if we wanted these ordered from most to fewest applications? Add `order by count(*) DESC` to end of query.

   What if we wanted to "drill down" and count number of applications to each major at each college? Replace `cName` with `cName, major` in both select and group by clauses.

   What if we wanted to count number of *applicants* rather than applications? Replace `count(*)` with `count (distinct sID)`.

8. **Filtering aggregated rows** Suppose we wanted only schools that admitted at least 3 applications. We *cannot* use the where clause because the where clause is evalued *before* rows are grouped. Instead we must use the having clause.

   ```
   select cName, count(*) as numAccepted
   from Apply
   where decision = 'Y'
   group by cName
   having count(*) >= 3;
   ```

9. **Joining relations** To combine multiple relations, place relation names in from clause and use where clause to filter the results. Return the states in which colleges are located that have admitted CS majors. Why the duplicates?

   ```
   select state
   from apply a, college c
   where a.cName = c.cName and major = 'CS' and decision = 'Y'
   ```