Schema for examples:

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

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

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

2. **Filter rows** Use the where clause to filter the results. Use parentheses to control the order in which the expression is evaluated (analogous to math: $5 \times (3 + 4) = 35$ whereas $5 \times 3 + 4 = 19$). Find applications where the major is CS and the college is either Colgate or Bucknell.

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

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

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

4. **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 ASC;
   ```

   The default is *ascending* order (A, B, C,..., Z). To order in *descending* order (Z, Y, X, ..., A), replace `order by cName ASC` with `order by cName DESC`.

5. **Aggregations** Use *aggregation functions* to summarize attribute values of collections of tuples. Aggregation functions include: COUNT, MIN, MAX, SUM, AVG. Find average GPA of all students.

   ```
   select avg(gpa) as averagegpa
   from student;
   ```

   Find the GPA spread (max to min) across all students from high schools with fewer than 1000 students.

   ```
   select max(gpa) - min(gpa)
   from student
   where sizehs < 1000;
   ```

6. **Groups and aggregations** Use the group by clause to group together rows with identical values in specified columns. If an attribute appears in the select clause, one the following conditions must be true: the attribute also appears in the group by clause, or the attribute value is aggregated.

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)`.

7. **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;
```

8. **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'
```

9. **Subqueries** The answer to every query is a relation. You can therefore compose a query on the result of a previous query. Find average GPA of students who applied to be CS majors.

```
select avg(gpa)
from student, (select sid from apply
               where major = 'CS'
               group by sid) as csmajors
where student.sid = csmajors.sid;
```

Alternatively, you can use the WITH clause:

```
with csmajors(sid) as (select sid from apply
                       where major = 'CS'
                       group by sid)
select avg(gpa)
from student, csmajors
where student.sid = csmajors.sid;
```