Queries

1) Tuple Relation Calculus (TRC)
   a. All queries in the form of: $\{T \mid p(T)\}$, to produce an output tuple T, p(T) must hold for that T
   b. p(T) is composed of:
      i. Atomic Formulas
         1. $R \in Relation$ R is a tuple from an existing relation
         2. $R.a \,\text{op}\, S.b$ Compare R.a to S.b
         3. $R.a \,\text{op}\, constant$ Compare R.a to a constant
      ii. Compound Formulas
         1. $\neg p, p \vee q, p \wedge q$ not p, p or q, p and q
         2. $\exists R(p(R))$ There exists at least one such R where p(R) holds true
         3. $\forall R(p(R))$ For every R that exists (in the universe), p(R) hold true
         4. $p \Rightarrow q$ is equivalent to $\neg p \vee q$
   c. Bound/Free Variables
      i. A variable is bound by using $\exists R(p(R))$ or $\forall R(p(R))$
      ii. All variables except the one output/answer variable must be bound
   d. Safe Programs - have a finite number of answers. Generally, negation can cause unsafe programs, i.e. $\{R \mid \neg(R \in Relation)\}$
2) Relational Algebra
   a. 5 basic operations
      i. σ - Selection – select a subset of rows from a relation (on the right)
      ii. π - Projection – select a subset of columns from a relation (on the right)
      iii. $\times$ - Cross Product – combine two relations, by computing all pairwise combinations of rows from the left relation with the rows from the right relation
      iv. $-$ - Set difference – Find tuples in the left relation that are not in the right relation
      v. $\cup$ - Union – combine tuples in the left and right relations
   b. Compound Operators
      i. $\cap$ - Intersection – $R \cap S = R - (R - S)$
      ii. $\triangleright\triangleleft$ - Join – $R \triangleright\triangleleft_c S = \sigma_c (R \times S)$
      iii. / - Division – $R / S = \pi_x(R) - \pi_x((\pi_x(R) \times S) - R)$

3) SQL

a. General Form:

subselect statement:

SELECT [ALL | DISTINCT] *expression* [**AS** *name*] {, *expression* [**AS** *name*]}
FROM *relation* [*name*] {, *relation* [*name*]}
[**WHERE** *search-conditions*]
[**GROUP BY** *column* {, *column*}]
[**HAVING** *search-conditions*]

full select:

subselect
{**UNION | INTERSECT | EXCEPT** [**ALL**] *subselect*}
[**ORDER BY** *result_column* [**ASC|DESC**] {, *result_column* [**ASC|DESC**]}]

b. Conceptual Evualation
   i. Compute the cartesion product of all tables (including views or other subselects) in the **FROM** clause
   ii. Discard rows not satisfying the **WHERE** *search-conditions*
   iii. Group the remaining rows such that each group has the same value for each column in the **GROUP BY** clause
   iv. Discard groups not satisfying the **HAVING** *search-conditions*
   v. Evaluate the expressions of the **SELECT** clause (and project out extra columns)
   vi. If **DISTINCT** is present, eliminate duplicate rows
   vii. After each subselect is evaluated, form the **UNION**, **INTERSECT**, **EXCEPT**
   viii. Order the remaining tuples by columns (and directions) as listed in the **ORDER BY** clause

c. *search-conditions* contains a predicate (or compound predicate) that evaluates to a boolean
   1. *boolean* **AND | OR** *boolean*
   2. NOT *boolean*
   3. *expression* op *expression* | *subselect*
   4. *expression* [**NOT**] **BETWEEN** *expression* **AND** *expression*
   5. *expression* **IS** [**NOT**] **NULL**
   6. *expression* **LIKE** *expression*
   7. *expression* [**NOT**] **IN** *subselect* | (*val* {*, val*})
   8. [**NOT**] **EXISTS** *subselect*
   9. *expression* **ANY** *subselect*
   10. *expression* **ALL** *subselect*

d. NULLs
   i. Three-values logic tables

OR	T	F	Null
T	T	T	T
F	T	F	Null
Null	T	Null	Null

AND	T	F	Null
T	T	F	Null
F	F	F	F
Null	Null	F	Null

NOT	
T	F
F	T
Null	Null

e. Joins
   i. INNER JOIN – only rows that match search-conditions are returned
   ii. LEFT OUTER JOIN – return all matched rows, and rows on the left that were unmatched (use nulls for other fields)
   iii. RIGHT OUTER JOIN – same as left, except return right rows that were unmatched
   iv. FULL OUTER JOIN – return all matched rows and rows that were unmatched
   v. NATURAL JOIN – most common join, equi-join for each pair of attributes with the same name