Relational Calculus
Calculus and Algebra

- Algebra: specifying how to obtain results
  - Procedural
  - SQL: specifying how to derive the results using the tables in the database

- What if a user does not know how to obtain the results?
  - Specifying what the results are can be easier than specifying how to get the results
  - Relational calculus specifies what instead of how
Tuple Relational Calculus

- A nonprocedural query language, where each query is of the form \( \{ t \mid P(t) \} \)
  - Results: the set of all tuples \( t \) such that predicate \( P \) is true for \( t \)
- \( t \) is a tuple variable, \( t[A] \) denotes the value of tuple \( t \) on attribute \( A \)
- \( t \in r \) denotes that tuple \( t \) is in relation \( r \)
- \( P \) is a formula similar to that of the predicate calculus
Predicate Calculus Formula

- A set of attributes and constants
- A set of comparison operators: (e.g., <, ≤, =, ≠, >, ≥)
- A set of connectives: and (\(\land\)), or (\(\lor\)), not (\(\neg\))
- Implication (\(\Rightarrow\)): \(x \Rightarrow y\), if \(x\) if true, then \(y\) is true  \(x \Rightarrow y \equiv \neg x \lor y\)
- A set of quantifiers
  - \(\exists t \in r (Q(t))\) \(\equiv\) ”there exists” a tuple in \(t\) in relation \(r\) such that predicate \(Q(t)\) is true
  - \(\forall t \in r (Q(t))\) \(\equiv\) \(Q\) is true “for all” tuples \(t\) in relation \(r\)
Banking Example

- branch (branch_name, branch_city, assets)
- customer (customer_name, customer_street, customer_city)
- account (account_number, branch_name, balance)
- loan (loan_number, branch_name, amount)
- depositor (customer_name, account_number)
- borrower (customer_name, loan_number)
Example Queries

• Find the loan_number, branch_name, and amount for loans of over $1200
  \[ \{ t \mid t \in loan \land t[amount] > 1200 \} \]

• Find the loan number for each loan of an amount greater than $1200
  \[ \{ t \mid \exists s \in loan (t[loan_number] = s[loan_number] \land s[amount] > 1200) \} \]
  – A relation on schema [loan_number] is implicitly defined by the query
Example Queries

• Find the names of all customers having a loan, an account, or both at the bank
  \[ \{ t \mid \exists s \in borrower ( t [customer_name] = s [customer_name]) \lor \exists u \in depositor ( t [customer_name] = u [customer_name]) \} \]

• Find the names of all customers who have a loan and an account at the bank
  \[ \{ t \mid \exists s \in borrower ( t [customer_name] = s [customer_name]) \land \exists u \in depositor ( t [customer_name] = u [customer_name]) \} \]
Example Queries

• Find the names of all customers having a loan at the Perryridge branch

\[
\{ t \mid \exists s \in \text{borrower} \ (t[\text{customer\_name}] = s[\text{customer\_name}] \land \exists u \in \text{loan} \ (u[\text{branch\_name}] = \text{"Perryridge"} \land u[\text{loan\_number}] = s[\text{loan\_number}]) \} \]

Example Queries

- Find the names of all customers who have a loan at the Perryridge branch, but no account at any branch of the bank

\[
\{ t \mid \exists s \in \text{borrower} \ (t \text{[customer_name] = s} \land \exists u \in \text{loan} \ (u \text{[branch_name] = “Perryridge”} \land u \text{[loan_number] = s} \land \text{not } \exists v \in \text{depositor} \ (v \text{[customer_name] = t} \text{[customer_name]})}\}
\]
Example Queries

• Find the names of all customers having a loan from the Perryridge branch, and the cities in which they live

\[ \{ t \mid \exists s \in loan \ (s[\text{branch\_name}] = \text{“Perryridge”} \land \exists u \in borrower \ (u[\text{loan\_number}] = s[\text{loan\_number}] \land t[\text{customer\_name}] = u[\text{customer\_name}]) \land \exists v \in customer \ (u[\text{customer\_name}] = v[\text{customer\_name}] \land t[\text{customer\_city}] = v[\text{customer\_city}])\} \]
Example Queries

• Find the names of all customers who have an account at all branches located in Brooklyn

\[ \{ t \mid \exists r \in \text{customer} ( t[\text{customer}_\text{name}] = r[\text{customer}_\text{name}]) \land ( \forall u \in \text{branch} ( u[\text{branch}_\text{city}] = \text{“Brooklyn”} \Rightarrow \exists s \in \text{depositor} ( t[\text{customer}_\text{name}] = s[\text{customer}_\text{name}] \land \exists w \in \text{account} ( w[\text{account}_\text{number}] = s[\text{account}_\text{number}] \land ( w[\text{branch}_\text{name}] = u[\text{branch}_\text{name}]))) ) ) \} \]
Safety of Expressions

• \{ t \mid \neg t \in r \} results in an infinite relation if the domain of any attribute of relation r is infinite
  – It is possible to write tuple calculus expressions that generate infinite relations
• To guard against the problem, we restrict the set of allowable expressions to safe expressions
Safe Expressions

- An expression \( \{ t \mid P(t) \} \) in the tuple relational calculus is safe if every component of \( t \) appears in one of the relations, tuples, or constants that appear in \( P \)
  - More than just a syntax condition
  - \( \{ t \mid t[A] = 5 \lor \text{true} \} \) is not safe --- it defines an infinite set with attribute values that do not appear in any relation or tuples or constants in \( P \)
Domain Relational Calculus

- A nonprocedural query language equivalent in power to the tuple relational calculus
  - Each query is an expression of the form
    \[ \{ <x_1, x_2, \ldots, x_n> | P(x_1, x_2, \ldots, x_n) \} \]
  - \( x_1, x_2, \ldots, x_n \) represent domain variables
  - \( P \) represents a formula similar to that of the predicate calculus
Example Queries

• Find the loan_number, branch_name, and amount for loans of over $1200
  \( \{ <l, b, a> | <l, b, a> \in \text{loan} \land a > 1200 \} \)

• Find the names of all customers who have a loan of over $1200
  \( \{ <c> | \exists l, b, a (<c, l> \in \text{borrower} \land <l, b, a> \in \text{loan} \land a > 1200) \} \)
Example Queries

• Find the names of all customers who have a loan from the Perryridge branch and the loan amount

  – \{ <c, a> | \exists l ( <c, l> \in borrower \land \exists b ( <l, b, a> \in loan \land b = "Perryridge") ) \}

  – \{ <c, a> | \exists l ( <c, l> \in borrower \land <l, "Perryridge", a> \in loan) \}
Example Queries

• Find the names of all customers having a loan, an account, or both at the Perryridge branch
  \[ \{< c > | \exists l (< c, l > \in borrower \land \exists b,a (< l, b, a > \in loan \land b = \text{"Perryridge"}) ) \lor \exists a (< c, a > \in depositor \land \exists b,n (< a, b, n > \in account \land b = \text{"Perryridge"}) ) \} \]

• Find the names of all customers who have an account at all branches located in Brooklyn
  \[ \{< c > | \exists s,n (< c, s, n > \in customer) \land \forall x,y,z (< x, y, z > \in branch \land y = \text{"Brooklyn"}) \Rightarrow \exists a,b (< x, y, z > \in account \land < c,a > \in depositor) \} \]
Safety of Expressions

• The expression \(< x_1, x_2, ..., x_n > \mid P(x_1, x_2, ..., x_n)\) is safe if all of the following hold:
  – All values that appear in tuples of the expression are values from \(\text{dom}(P)\) (that is, the values appear either in \(P\) or in a tuple of a relation mentioned in \(P\))
  – For every “there exists” subformula of the form \(\exists x \ (P_1(x))\), the subformula is true if and only if there is a value of \(x\) in \(\text{dom}(P_1)\) such that \(P_1(x)\) is true
  – For every “for all” subformula of the form \(\forall x \ (P_1(x))\), the subformula is true if and only if \(P_1(x)\) is true for all values \(x\) from \(\text{dom}(P_1)\)
Summary

• Relational calculus
  – An alternative query language
  – Specifying what instead of how
• Tuple relational calculus
• Domain relational calculus
To-Do-List

• Read Chapters 5.1 and 5.2 in the textbook
• Rewrite the queries in relational algebra using relational calculus