QUEL

- Tuple calculus-based language developed for INGRES (Interactive Graphics and Retrieval System)
  - DDL and DML: interactive query language as well as embedded within a host programming language
  - functionality similar to that provided by early SQL
- Most queries are expressed using two types of clauses
  1. declarations of tuple variables
     range of \( t_1 \) is \( r_1 \)
        \( t_1 \): tuple variable
        \( r_1 \): relation name
        \( t_1 \) is restricted to take values of tuples in \( r_1 \)
  2. retrieval
     retrieve (target list)
     where (conditions)
A Query Expression in QUEL

range of \( t_1 \) is \( r_1 \)
\[
\vdots
\]
range of \( t_m \) is \( r_m \)
retrieve \((t_{i_1}.A_{j_1}, ..., t_{i_n}.A_{j_n})\)
where \( P \)

\[
\equiv \{ u \mid (\exists t_1 \in r_1) \ldots (\exists t_m \in r_m) \\
[u(A_{j_1})=t_{i_1}(A_{j_1}) \land \ldots \land u(A_{j_n})=t_{i_n}(A_{j_n}) \land P(t_1, ..., t_m)] \}
\]

range of \( t \) is deposit
retrieve \((t.C-name)\)
where \( t.B-name="Perry" \)

range of \( t \) is deposit
range of \( s \) is loan
retrieve \((t.C-name)\)
where \( t.B-name="Perry" \) and \( s.B-name="Perry" \)
and \( s.C-name=t.C-name \)
Features of QUEL

- No nested subqueries are allowed
  (no nested retrieve-where clause inside a where clause)

- Name change and assignment of results
  - attributes in the resulting relation can be changed
    range of s is customer
    retrieve (doubleuser = s.C-name)
  - retrieve into <relation name> (target list)
    range of u is deposit
    retrieve into temp (u.C-name)
    where u.B-name = "Perry"

- Duplicate elimination must be explicitly requested
  retrieve unique <target list>
Features of QUEL

- Tuple variables
  - useful when the two distinct tuple variables from the same relation is referred

  range of u is deposit
  range of s is deposit
  retrieve unique (s.C-name)
  where u.C-name="Jones" and u.C-city=s.C-city

- Aggregation operators: count, avg, min, max, sum
  countu, sumu, avgu → eliminate duplicates

  range of t is account
  range of u is account
  retrieve (t.Account-number)
  where t.Balance > avg(u.Balance)
Features in QUEL

- **Data and storage definition**
  - create `<relation name>` (attribute, domain)
  - modify to specify or change storage structure:
    ISAM, hash, Btree, Heap, C(ompressed)
  - index on `<relation name>` is `<index name>` (attributes)

- **Update operations**
  - append to
    append to account (B-name="Perry",
    A-number=A1, balance=500)
  - delete
    range of t is deposit
    delete t
    where t.C-name="Jones"
  - replace
    range of t is deposit
    replace t (balance = 1.05*balance)
Features in QUEL

- Comparison with SQL
  - QUEL does not have explicit set operations such as union and intersection, but can implement them

Q: Find all customers who have account, loan, or both at Perry

range of u is deposit
retrieve into temp (u.C-name)
where u.B-name="Perry"
range of s is borrow
append to temp (s.C-name)
where s.B-name="Perry"
range of t is temp
retrieve unique (t.C-name)

- Other operators
  - where any ... explicit existential quantifier
  - sort by
  - define view
Query-By-Example (QBE)

- Based on domain calculus
  - developed by Zloof at IBM T. J. Watson Lab, and available as QMF interface option to IBM DB2
  - formulates query by filling in templates to construct an example, using skeleton tables
  - variables are existentially quantified and preceded by the underscore character: _X, P._Y (P for print)
  - strings without underscore are constants: Smith
  - blank columns represent don’t care
  - same variable appearing in two separate table is a join attribute
  - duplicates are eliminated automatically
  - or condition is represented by providing a separate example row
Features in QBE

- Condition box
  - to express constraints
  
  Borrow: B-name: Loan#: C-name: Amount
  
  _X _Y P._Z _a

  Conditions
  
  X ≠ Perry
  a > 1000

- Insert (I), delete (D), update (U)

  Borrow: B-name: Loan#: C-name
  
  D. Perry 1234

- Aggregate operators

  - AVG, MAX, MIN, CNT, SUM
  
  - to allow all the values are considered, all the aggregate functions are postfixed with ALL