11 – Networking

1. Database Access API

   A. Open DataBase Connectivity

      i. Standard API for DB access

      ii. Based on “device driver” model
      ODBC driver converts query to DBMS-specific calls

   B. Java DataBase Connectivity

      i. Standart API for DB access by Java program(Not by C)

      ii. Considered as ODBC for Java

      iii. Approach

         1. JDBC-ODBC bridge (use ODBC)

         2. Native-API driver (use native DBMS API)

         3. Network-Protocol driver (use middleware)

         4. Database-Protocol Driver (self-convert and call)

2. Database Network Protocols

   A. Use API over TCP/IP

   B. Process :
   authentication – request – execute query – serialize results and reply

   C. Using protocols that already exists has benefits:
   reuse client drivers,
   no need to develop network protocols.

D. Design Decision

   i. Row vs. Column Layout
   ODBC, JDBC is row oriented protocols
   for analytical query, send data in vector

   ii. Compression
   optimal compression weight is varies on network speed.

   iii. Data Serialization
   binary encoding (cheap overhead if DB's format is close to binary)
   or text encoding (no need to consider endianness)

   iv. String Handling
   null termination(mark the end of string),
   length-prefixes(mention string length)
   or fixed width(padding)

3. Kernel Bypass Methods

   A. OS's TCP/IP stack is slow because of overhead of context switch and interrupt, data copying, lots of latches

   B. Data Plane Development Kit

      i. Libraries that allows accessing Network Interface Controller directly

      ii. No system call or data copy, because of direct access to NIC buffer

   C. Remote Direct Memory Access

      i. Read and write memory directly on a remote host
      no going through OS

      ii. Client should know exact address of data
      server doesn't know that memory is accessed by other machines