TM-0085

Delta Demonstration at ICOT Open House

Kunio Murakami, Takeo Kakuta, Nobuyoshi Miyazaki,
Shigeki Shibayama, Haruo Yokota
and Delta Demonstration Team

November, 1984

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

Technical Memorandum of ICOT Research Center: TM-0085

Delta Demonstration at ICOT Open House

November 1984

Kunio Murakami, Takeo Kakuta, Nobuyoshi Miyazaki,
Shigeki Shibayama, Haruo Yokota,
and Delta demonstration team

ICOT Research Center
Institute for New Generation Computer Technology
Mita Kokusai Bldg. 21F, 1-4-28 Mita,
Minato-ku, Tokyo, 108
Phone: 03-456-3195, Telex: ICOT J32964

Preface

This document is a collection of materials used in the Delta demonstration at ICOT Open House on November 12-14, 1984. The materials have been edited according to the "outline of the demonstration", which is included as the first page of this document. Queries and their results shown on the display screens are reproduced. However, we have not been able to reproduce actual maps displayed on screen due to the limitation of the printing facilities. Thus, maps in this document lack certain information such as location symbols and resorts names. Moreover, resultant maps of some queries are not shown.

The demonstration was conducted by the Delta demonstration team. The team consisted of the researchers and engineers of ICOT, Toshiba Corporation, Hitachi Ltd., and Oki Electric Industry Co., Ltd. Members of the team are as follows:

(1) Supervising
Kunio Murakami (ICOT),
Akira Miyoshi (Toshiba),
Akira Tachigami (Hitachi)

(2) Setting, Maintenance and Operation
Takeo Kakuta, Akihito Taguchi (ICOT),
Kazunori Shimakawa, Masayuki Murakami, Masayuki Mihei,
Masato Yamashita, Tetsuji Mori, Ikuo Akada (Toshiba),
Hiromichi Ishikawa, Keiko Suzuki, Masaru Shima,
Isamu Matsushima, Tatsuo Yajima (Hitachi),
Yukihiro Morita, Kazunori Kojima (Oki)

(3) Explanation
Nobuyoshi Miyazaki, Shigeki Shibayama (ICOT),
Kazuhide Iwata, Susumu Matsuda, Yasuo Hoshino (Toshiba),
Kazuhiko Omachi, Kazuhiro Sato, Takuo Ishizuka (Hitachi)

(4) User (Host operation)
Haruo Yokota (ICOT),
Hiroshi Sakai, Masaaki Abe, Yasuo Toyomoto,
Takayoshi Fushinuki (Toshiba).

The contents of this documents were written by members of ICOT, Explanation group and user group.

December 1984,
Editor: N. Miyazaki.

Outline of the Delta Demonstration

November 1984

INSTITUTE FOR NEW GENERATION COMPUTER TECHNOLOGY

Welcome to the Delta Demonstration! The contents of the demonstration are as follows.

1. Explanation of Delta

We explain Delta using panels and recorded narration. There are six panels entitled as follows:

(1) Development Objectives of RDBM Delta
(2) Pilot System for Software Development
(3) Delta Architecture
(4) Example of Query Processing
(5) Query Processing Flow
(6) Delta Configuration

2. Explanation of the Demonstration

Two systems, Interactive Query Language system (IQL) and Inference/Relational-database Interface System (IRIS), are demonstrated using a "sightseeing" database.

The "sightseeing" database consists of six relations (tables). They are "resort," "recreation," "hotel," "resort-map," "location," and "neighbor." The database contents are picked up from sightseeing guidebooks in Japan. (Appendix A)

3. Demonstration of IQL

The demonstration of IQL shows a usage of relational databases. A series of queries is executed according to a scenario. (Appendix B)

4. Demonstration of IRIS

IRIS demonstrates how to use relational databases in logic programming. IRIS provides a system to combine Prolog and relational databases. Users give Prolog queries to the system. The system answers using information in Prolog and Delta. "Neighbor" relation is used in this demonstration.

5. Questions and Answers

1. Explanation of Delta

Hello! Welcome to the Delta demonstration. Here, we would like to show you some sample sessions with Delta. First, please look at the panels around you. We will briefly explain some of Delta's features using these panels before we begin the sample sessions.

Panel 1 summarizes the objectives of developing Delta. It represents a part of the effort to develop knowledge base machines at ICOT. We decided to develop knowledge base machines in a step-by-step manner, because the functions and characteristics required for knowledge base machines are not fully understood. The basic functions of knowledge base machines should include the manipulation of knowledge and data, and should support inference using these information. The inference function is being studied by other ICOT groups. Our group's main objective is to develop architectures capable of handling a large amount of knowledge and data. We selected a relational database machine as the starting point because of its compatibility with the logic programming languages. As you know, the basic vehicle for FGCS project is the logic programming. The relational database model is based on first-order predicate logic, and its correspondence to logic programming languages is well known.

Thus, the objectives of Delta can be summarized as follows:
First, to create an experimental environment for various knowledge base operations.

Second, to create a software development tool by connecting Delta with Personal Sequential Inference machines, PSIs for short, via a LAN.

Third, data collection for knowledge base machines using Delta.

Panel 1

Development Objectives of RDBM Delta

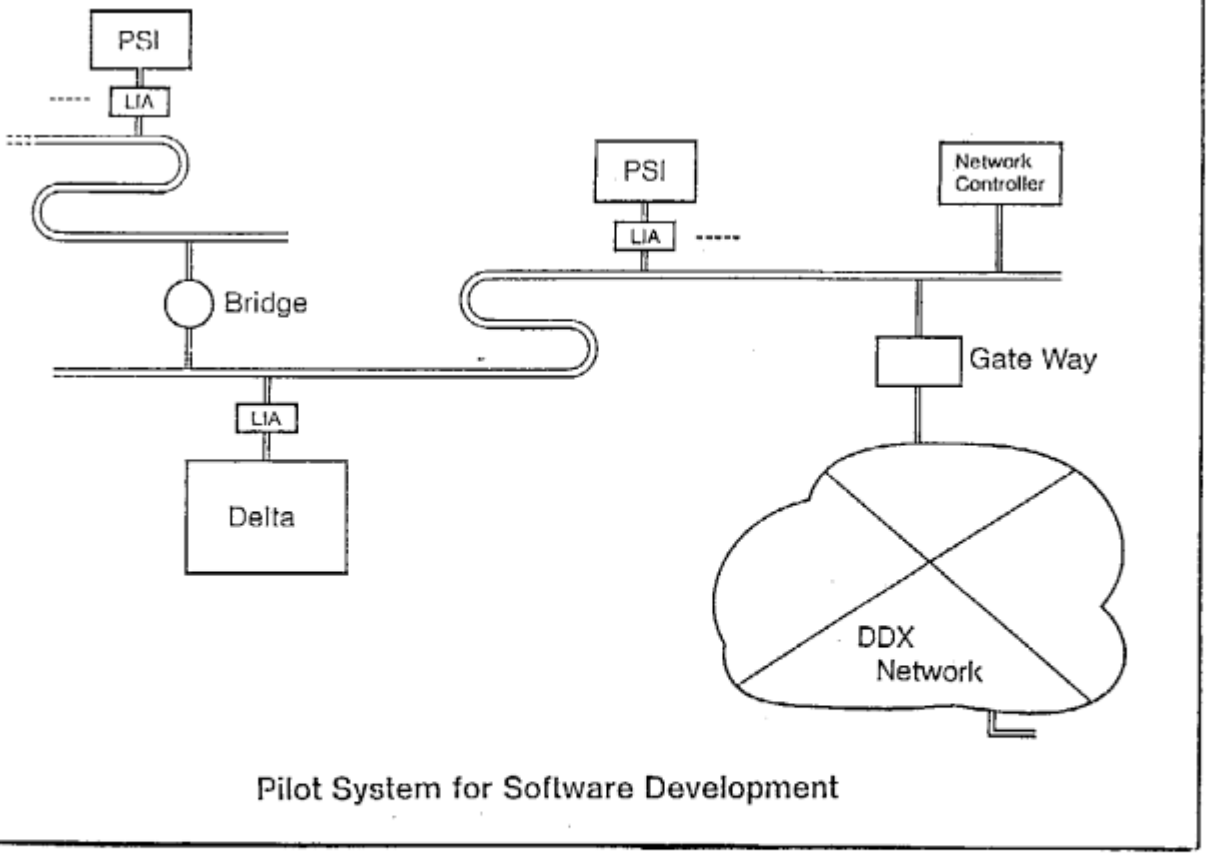
1. To create an experimental environment for various knowledge base operations.

2. To create a software development tool connecting Delta with PSIs via a LAN.

3. Data collection toward KBM using Delta
   (1) Experiments Delta tightly-coupled with PSI.
   (2) Parallel processing experiments of queries using up to four REs.

Next, panel 2 shows the total system environment being developed at ICOT for developing the software that will be used in future research. The system will consist of a local area network, a number of PSIs, and Delta. The LAN consists of several unit-networks connected by Bridges. Unit-networks are similar to Ether-net. LAN Interface Adapters, LIAs for short, provide a high-level interface to terminals such as PSIs and Delta. LIAs support a group communication protocol, as well as the more common session-based protocol. The LAN will be connected to a public packet-switched network, DDX, by a gateway. Currently, one unit-network is operational.

Panel 2



Pilot System for Software Development

Panel number 3 explains Delta's Architecture. Delta has a functionally-distributed, multi-processor configuration. The access command interface is based on relational algebra. Delta has dedicated hardware for relational algebra operations. Its internal schema is attribute-based instead of tuple-based, and a two-level clustering method is applied to reduce the amount of data to be accessed. Furthermore, it adopts a hierarchical memory subsystem with a large capacity memory to reduce the number of accesses to the secondary storage.

Panel 3

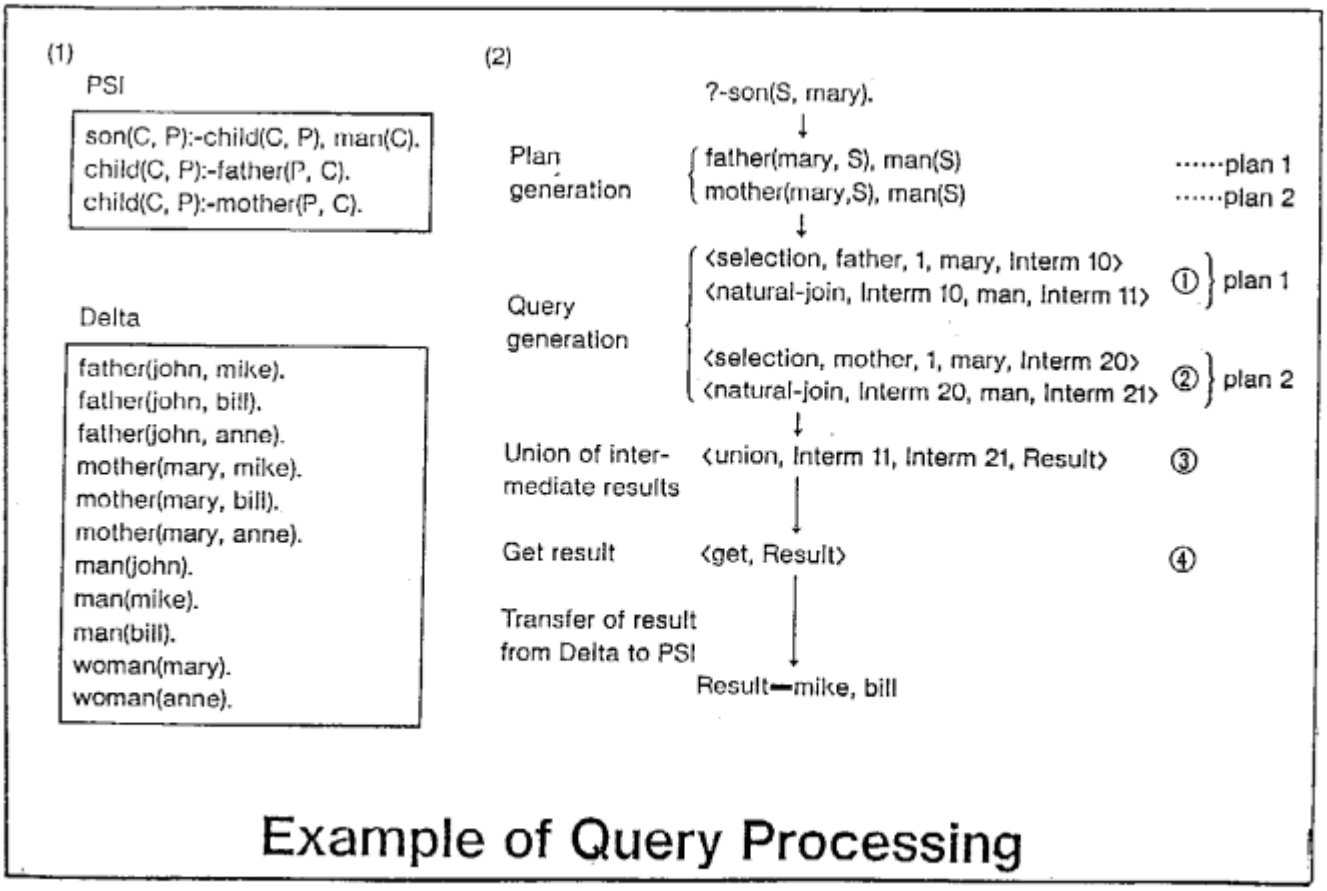
Delta Architecture

1. Functionally distributed multiprocessor configuration
2. Relational algebra command interface with hosts
3. Dedicated hardware for relational algebra operations
4. Attribute-based internal schema, two-level clustering method
5. Interface for two-channel stream data transfer : RE-HM
6. Hierarchical Memory with large semiconductor memory unit
7. Statistical information collection function

Panel number 4 illustrates query processing by PSI and Delta. A "family" database is used in this example. This shows one way to use a relational database with Prolog. PSI has been supplied with a number of rules that define family relations. For instance, one rule says that a parent is either a father or a mother. On the other hand, Delta stores a number of facts as relations expressing individual relationships between family members.

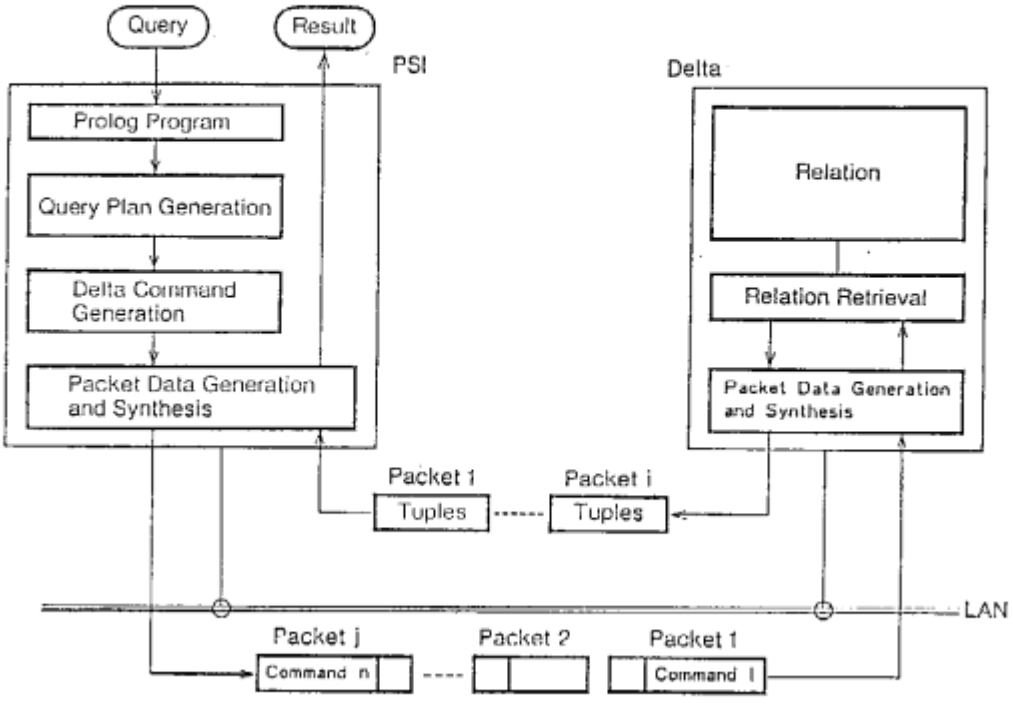
Suppose we use Prolog to ask PSI for the names of Mary's sons. PSI first analyses this query using its rules and generates plans to obtain the desired information from Delta. It then translates the query into sets of Delta access commands. It may generate several such command sets for a single query. PSI then sends these command sets to Delta. After the results are computed in Delta, they are sent back to PSI. This process is also outlined in panel 5. It shows how PSI and Delta interact via the LAN.

Panel 4



Example of Query Processing

Panel 5

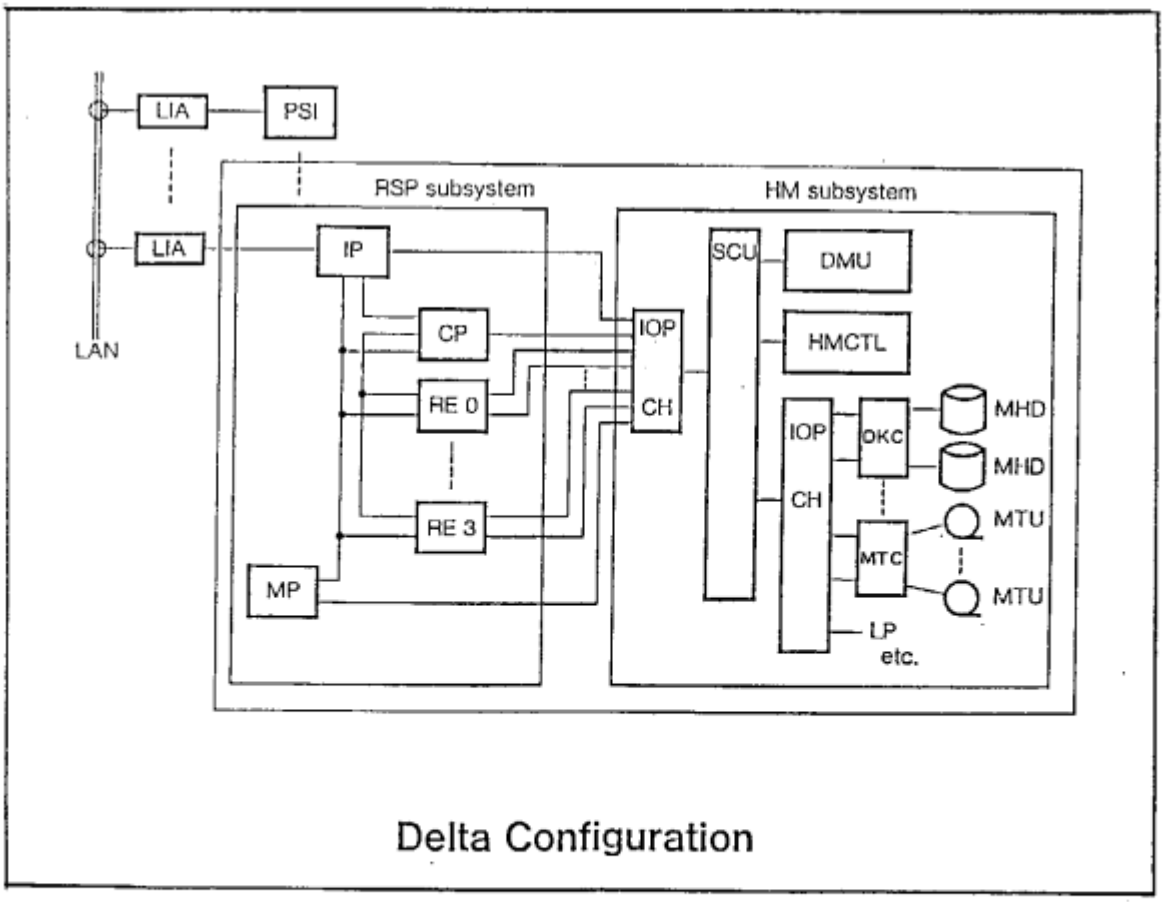


Query Processing Flow

Panel 6 shows the hardware configuration of Delta. It consists of two subsystems, the RDBM Supervisory and Processing subsystem, RSP for short, and the Hierarchical Memory subsystem, HM.

The RSP subsystem consists of four major components: an Interface Processor (IP), a Control Processor (CP), Relational database Engines (REs) (of which there will be four), and a Maintenance Processor (MP). The IP is a kind of front-end processor. The CP acts as the main controller for Delta. It controls transactions, translates queries into internal format and directs the REs and HM during query processing. The REs are dedicated hardware for relational operations. They have pipelined two-way merge sorters to presort data for relational operations. Currently, only one RE is installed.

Panel 6



Delta Configuration

The HM consists of three main components: the HM controller, a database memory unit, and moving-head disks. The HM controller is a general-purpose processor. The database memory unit is a large semiconductor memory, that is used for the Delta's main working buffers and for a disk cache area. The capacity of the database memory unit is currently 16 megabytes, but will be expanded to 128 megabytes at the end of March, next year. Moving head disks are the secondary storage devices that contain the actual database. The capacity of each disk unit is about 2.5 gigabytes. The system has two disk units now, and will eventually have 8 units. Delta has a battery back-up unit to protect the database, particularly the contents of the database memory unit, against power failures.

Now, we have briefly explained Delta and the environment in which it is used. You can find more information in the FGCS Proceedings.

If you will please be seated, we will demonstrate some sample sessions with Delta.

2. Explanation of the Demonstration

First, we would like to explain the kind of sessions you are about to see. The demonstration system consists of Delta and a small host computer connected by a LAN. Two sessions will be displayed, both making use of a sightseeing database. The first session demonstrates the usual usage of a relational database using a query processor written in Prolog. The second session demonstrates one way to connect Prolog with a relational database: this combination constitutes a deductive database system.

Please look at the white board on your right. As you can see, the sightseeing database consists of six relations.*

The "resort" relation contains name of the resort, its prefecture, distance from Tokyo, mode of transportation and fare required to reach it, and local souvenirs. A prefecture in Japan is

---

* See appendix A.

something like a state in the United States.

The "recreation" relation includes information about activities you can participate in at particular resorts.

The "hotel" relation provides information about resort accomodations.

Two more relations, "resort-map" and "location", include maps showing the location of resorts.

Finally, the "neighbor" relation contains the distances and directions between resorts.

The first session uses all these relations except the last one, "neighbor". The second session uses only the "neighbor" relation. The query language of the first session, Interactive Query Language, or IQL, is similar to SEQUEL but is enhanced with the capability of displaying map information. The meanings of the IQL queries should be fairly obvious. The second session demonstrate IRIS, an Inference/Relational-database Interface System. IRIS combines Prolog with a relational database. Users input ordinary queries in Prolog, and the system obtains the requested information from Delta.

We may request any basic information about sightseeing in Japan in these sessions. However, we use a sequence of queries based on a fixed scenario so as to complete the sessions in a reasonable amount of time. You may find the scenario in appendix B of your leaflet. It starts as follows:

A famous database designer has come to Japan to participate in the International Conference on Fifth Generation Computer Systems, 1984. Using this opportunity, he plans to go sightseeing around Tokyo. However, he does not have enough information about resorts in Japan.

When he visits ICOT, he finds the relational database

machine, Delta. Fortunately, it includes a database of Japanese resorts. He asks an ICOT researcher to use Delta to search for appropriate resorts. The researcher willingly accepts his request.

.....

The demonstration will be conducted by the people of the Delta design team. We should warn you that, as Delta will not be fully operational until March of next year, its performance can at times be somewhat erratic. We hope you will bear with us if Delta decides to be uncooperative!

3. Demonstration of IQL

Good morning (afternoon) everybody. Now we start the demonstration sessions. You may have questions, but please wait until the end of the demonstration. By the way, the first piece of Delta hardware is around you. Behind you is the RSP subsystem. From left to right, they are maintenance processor, interface processor, control processor, and a relational database engine. And, the Hierarchical Memory subsystem consists of these equipments. The HM controller and the database memory unit are over there. The moving-head disk units are behind these boxes.

In IQL session, this screen shows you the queries and compiled requests to Delta. The other one, over there, displays maps and resort locations. Maps themselves are stored in "resort-map" relation in Delta. For some queries, the results are listed here in tables. For others, results are displayed in a map over there.

--------- First Query ----------

Now, the first query please.

```
IQL>display.
IQL>select map_data.
IQL>from resort_map.
IQL>where area_name=[honshu].
```

Green letters express queries in IQL. This query directs the system to get map of "honshu" from Delta and display it on that screen. "Honshu" is the largest island of Japan.

Delta commands please.

Blue letters show Delta commands. Delta command is the interface of Delta with hosts. It is based on relational algebra.

The first command is "selection", the second is "projection" to get only necessary attribute. These two form a command-tree that specifies the retrieval. The third command demands to send the result to host.

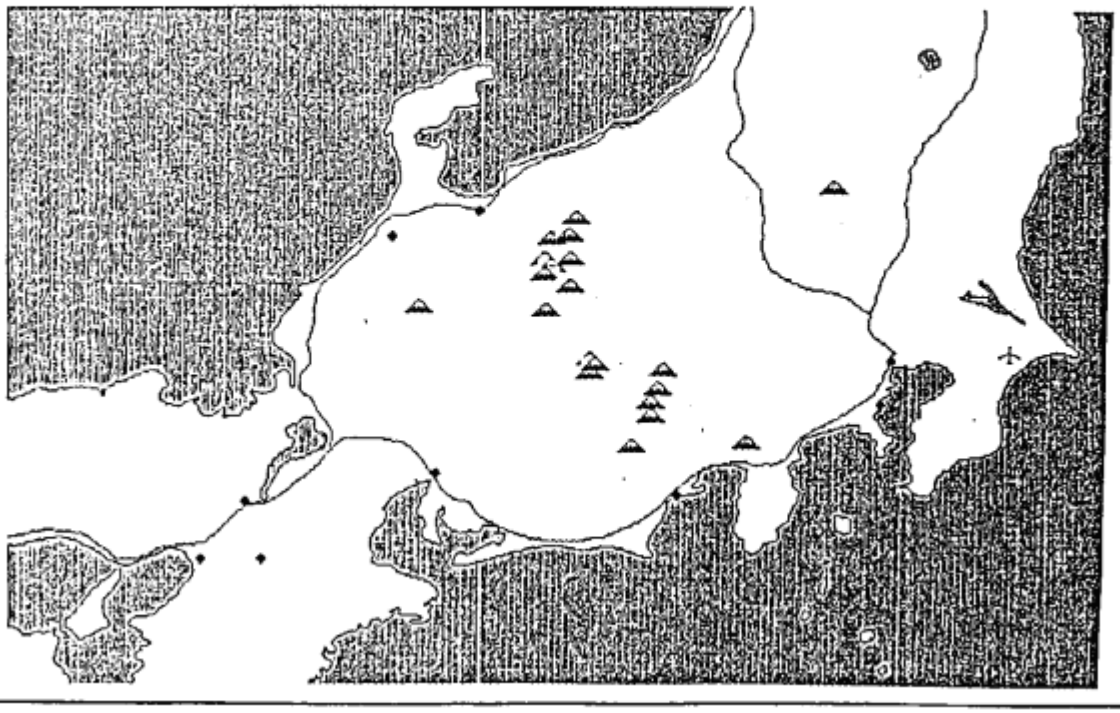
We got a map of central Japan on screen.

```
IQL>display.
IQL>select map_data.
IQL>from resort_map.
IQL>where area_name=[honshu].
IQL>
```

```
[[1,sel,p,840926120000,resort_map,d,1='686f6e736875'.*],

   [2,pro,t,1,1,2,ir35]]

[[1,get,i,ir35]]
```



---------- 2nd Query ----------

```
IQL>locate.
IQL>select symbol.
IQL>from location.
IQL>where area_name=[honshu] and resort_name=next.
IQL>select name.
IQL>from resort.
IQL>where distance>[100] and distance<[150].
```

The user wants to go out of Tokyo, but not too far. He directs the system to display resorts that are between 100km and 150km from Tokyo. The first line, "locate", means to locate results in the map.

Commands please.

This query has nested selections involving two relations, "location" and "resort". Thus, resultant commands include "semi-join" that is one kind of "join". Delta can process several kinds of "join"; namely "natural-join", "theta-join" and "semi-join".

We got the results of the query blinking on the map.

There are two major areas that have several blinking resorts. One is around "nikko" in north and the other is around "izu".

He prefers the place near sea.

(The result is not included in this document).

```
IQL>locate.
IQL>select symbol.
IQL>from location.
IQL>where area_name=[honshu]and resort_name=next.
IQL>select name.
IQL>from resort.
IQL>where distance>[100]and distance<[150].
IQL>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 [[1,sel,p,840810120000,resort.d,3>'0064'&3<'0096',*],

    [2,sel,p,840926120000,location.d,2='686[6e736875',*],

    [3,sjo,t,2,t,1,d,1,1,*],

    [4,pro ,t,3,1,3,ir36]]

 [[1,get,i,ir36]]
```

---------- 3rd Query ----------

```
IQL>select name,distance,transportation,souvenir1.
IQL>from resort.
IQL>where distance>[100] and distance<[150] and prefecture=[shizuoka].
IQL>order by distance.
```

He asks the system the name of the resorts, their distances, transportation from Tokyo, and souvenirs he can get there. The result should be sorted by their distances.

Delta commands please.

He got lots of resorts listed on screen.

```
IQL>select name,distance,transportation,souvenir1.
IQL>from resort.
IQL>where distance>[100]and distance<[150]and prefecture=[shizuoka].
IQL>order by distance.
IQL>
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
[[1,sel,p,840810!20000,resort,d,3>'0064'&3<'0096'
      &2='7368697a756f6b61',*],
   [2,pro,t,1,4,1,3,4,6,*],
   [3,srt,t,2,1,a,2,ir37]]

[[1,get,i,ir37]]
```

name	distance	transportation	souvenirl
nirayama	101	jnr	strawberry
ito	102	jnr	dried_fish
hokkawa	105	jnr-izukyu	dried_fish
izunagaoka	105	jnr	bun
numazu	105	jnr	sashimi
mishima	108	jnr	sashimi
shuzenji	110	jnr-izuhakone	horse-radish
yoshina	113	jnr-izuhakone	brawn
tsukigase	114	jnr-izuhakone	ayu
atagawa	116	jnr-izukyu	orange
funabara	118	jnr-izuhakone	brawn
amagi	119	jnr-bus	horse-radish
izukatase	119	jnr-izukyu	dried_fish
yugashima	119	jnr-bus	horse-radish
sagasawa	120	jnr-izuhakone	horse-radish
inatori	122	jnr-izukyu	dried_fish
toi	125	jnr-bus	sashimi
imaihama	126	jnr-izukyu	sillago
odaru	127	jnr-izukyu	crab
rendaiji	133	jnr-izukyu	bun
shimoda	135	jnr-izukyu	dried_fish
mihonomatsubara	137	jnr-bus	strawberry
izuosawa	138	jnr-izukyu	ayu
kumomi	138	jnr	sashimi
shimizu	138	jnr	green_tea
matsuzaki	142	jnr-bus	sashimi
yumigahama	143	jnr-izukyu	sillago
izudogashima	143	jnr-izuhakone	sashimi
umegashima	143	jnr-bus	bun
shimogamo	146	jnr-izukyu	melon
shizuoka	149	jnr	green_tea

---------- 4th Query ----------

```
IQL>display.
IQL>select map_data.
IQL>from resort_map.
IQL>where area_name=[izu].
```

This query asks for a more detailed map.

Delta commands please.

You may have noticed that relation names appear in Delta commands, but there are no attribute names in them. Delta distinguishes attributes by their places in a relation, just as Prolog distinguishes arguments by there places. So this (1=...) means the first attribute equals to ..... This ('697a75') is just "izu" expressed in hexa-decimal of ASCII code.

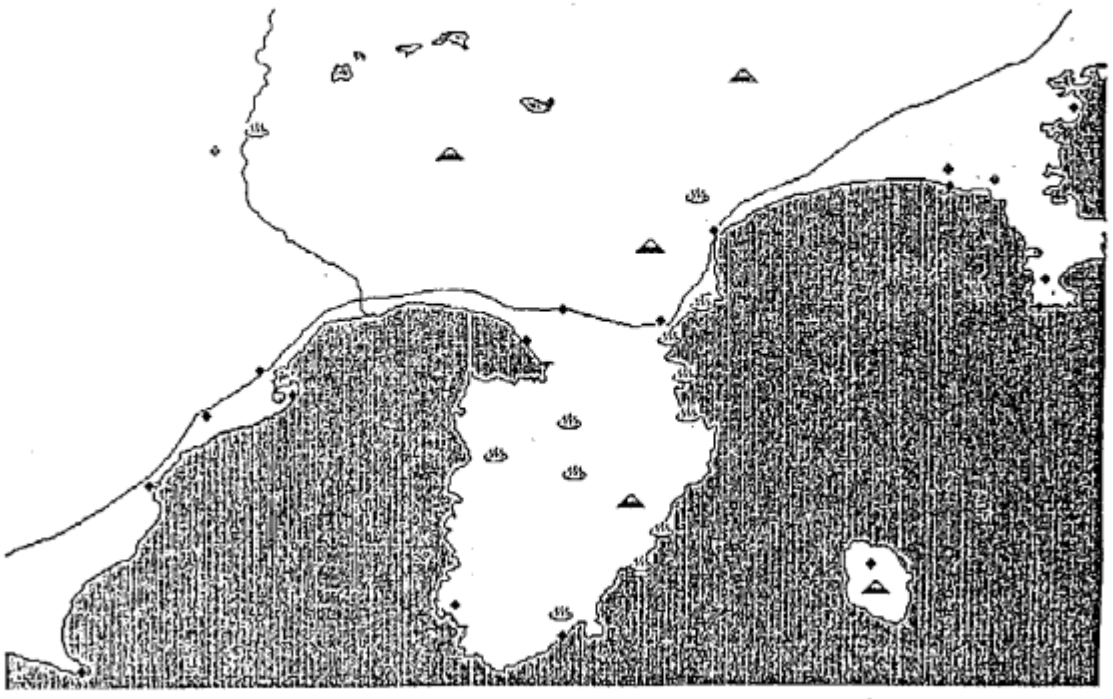
We got a map around "izu" on screen.

```
IQL>display.
IQL>select map_data.
IQL>from resort_map.
IQL>where area_name=[izu].
IQL>
```

```
[[1,sel,p,840926120000,resort_map,d,1='697a75',*],

   [2,pro,t,1,1,2,ir38]]

[[1,get,i,ir38]]
```



---------- 5th Query ----------

```
IQL>locate.
IQL>select symbol.
IQL>from location.
IQL>where area_name=[izu] and resort_name=next.
IQL>select name.
IQL>from resort.
IQL>where distance>[100] and distance<[150] and
    prefecture=[shizuoka] and souvenir1=[dried_fish].
```

The result should be shown on the map again.

Delta commands please.

Japan is known as a leader of fishing industry. So, He thinks it nice to have some kind of fish meal on tour. For souvenir, fish products are preferable to fresh fish. He looks for the dried fish for the souvenir by the recommendation of ICOT people.

This number (8408101200) in Delta command shows the date and time (August 10, 1984, 12:00:00) when the latest schema change of the "resort" relation was made. This information is stored in the dictionary in Delta, and hosts can read it by ordinary Delta commands. This information is attached here to ensure that the host uses correct dictionary.

Now, he gets candidate resorts to go on screen.
(The result is not included in this document).

```
IQL>locate.
IQL>select symbol.
IQL>from location.
IQL>where area_name=[izu]and resort_name=next.
IQL>select name.
IQL>from resort.
IQL>where distance>[100]and distance<[150]and
      prefecture=[shizuoka]and souvenir1=[dried_fish].
IQL>
```

```
[[1,sel,p,840810120000,resort,d,3>'0064'&3<'0096'
      &2='7368697a756f6b61'&6='6472696564f66697368',*],
   [2,sel,p,840926120000,location,d,2='697a75',*],
   [3,sjo,t,2,t,1,d,1,1,*],
   [4,pro,t,3,1,3,ir39]]
[[1,get,i,ir39]]
```

---------- 6th Query ----------

```
IQL>select resort-name,recreation-name,recreation-fee,resort-fare.
IQL>from resort,recreation.
IQL>where resort-name=recreation-resort_name and
    resort-distance>[100] and resort-distance<[150] and
    resort-prefecture=[shizuoka] and resort-souvenir1=[dried_fish].
```

He is asking next query (6th). He has not decided which kind of recreation he will enjoy there. So he asks the system about recreation over there.

Delta commands please.

This query involves joining "resort" and "recreation" relations.

This time the result is listed in a table. He decides to go to "shimoda".

He thinks he has heard the name of the resort before. ICOT people explain him that "shimoda" is the first Japanese port opened to foreign ships after nearly 300 years of closing ports in Edo era. He may have read about it in books.

```
IQL>select resort-name,recreation-name,recreation-fee,resort-fare.
IQL>from resort,recreation.
IQL>where resort-name=recreation-resort_name and resort-distance>[100]
    and resort-distance<[150]and resort-prefecture=[shizuoka]
    and resort-souvenir1=[dried_fish].
IQL>

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

[[1,sel,p,840810120000,resort,d,3>'0064'&3<'0096'&
       2='7368697a756f6b61'&6='6472696564f66697368',*],
   [2,joi,t,1,p,840810120000,recreation,d,1=2,*],
   [3,pro,t,2,4,1,9,0b,5,ir40]]
[[1,get,1,ir40]]
```

resort-name	recreation-name	recreatio	resort-fa
hokkawa	fishing	2	3
hokkawa	swimming	1	3
inatori	golf	30	3
inatori	tennis	5	3
inatori	gateball	2	3
inatori	baseball	2	3
inatori	swimming	1	3
inatori	fishing	2	3
inatori	field_athletics	2	3
ito	golf	30	2
ito	tennis	5	2
ito	bowling	2	2
ito	fishing	2	2
ito	swimming	1	2
ito	boating	1	2
ito	yachting	10	2
ito	wind_surfing	10	2
ito	cycling	1	2
ito	field_athletics	2	2
ito	riding	20	2
izukatase	golf	30	3
izukatase	tennis	5	3
izukatase	gateball	2	3
izukatase	baseball	2	3
izukatase	swimming	1	3
izukatase	wind_surfing	10	3
izukatase	cycling	1	3
izukatase	fishing	2	3
izukatase	field_athletics	2	3
izukatase	skin_diving	10	3
shimoda	skin_diving	10	3
shimoda	golf	30	3
shimoda	tennis	5	3
shimoda	gateball	2	3
shimoda	baseball	2	3
shimoda	swimming	1	3
shimoda	boating	1	3
shimoda	yachting	10	3
shimoda	wind_surfing	10	3
shimoda	cycling	1	3
shimoda	fishing	2	3
shimoda	field_athletics	2	3

---------- 7th Query ----------

```
IQL>select name,rate,capacity,accommodation.
IQL>from hotel.
IQL>where rate=<[15] and resort_name=[shimoda].
IQL>order by rate.
```

This query asks for information of hotels in shimoda. He wants to stay in a hotel not too expensive.

|The unit of the hotel rate is a 1,000 yen.

Delta commands please.

He got several results listed.

```
IQL>select name,capacity,rate,accommodation.
IQL>from hotel.
IQL>where rate=<[15]and resort_name=[shimoda].
IQL>order by rate.
IQL>
```

```
[[1,sel,p,840810120000,hotel,d,4=<'000f'&2='7368696d6f6461'.*],
   [2,pro,t,1,4,1,3,4,5,*],
   [3,srt,t,2,1,a,3,ir41]]
[[1,get,i,ir41]]
```

name	capacity	rate	accommodat
hotel_masaki	200	10	brkf/dnnr
shimoda_tokyu_hotel	380	12	brkf/dnnr
hamano_hotel	100	12	brkf/dnnr
aizuen	90	12	brkf/dnnr
ryokan_shimoda_juraku_hotel	290	12	brkf/dnnr
shimoda_grand_hotel	380	12	brkf/dnnr
hotel_izukyu	400	13	brkf/dnnr
shimoda_onsen_hotel	316	13	brkf/dnnr
shimoda_view_hotel	500	14	brkf/dnnr
kurofune_hotel	380	15	brkf/dnnr
shimoda_yamatokan	350	15	brkf/dnnr
yubatakan	150	15	brkf/dnnr
shimoda_kaihin_hotel	200	15	brkf/dnnr

---------- 8th Query ----------

```
IQL>select      address,phone.
IQL>from hotel.
IQL>where name=[shimoda_view_hotel].
```

He decides to stay at the largest hotel, shimoda-view-hotel, and wants to know its address and phone number.

Delta commands please.

We've got the result on screen. This finishes the IQL session.

```
IQL>select address,phone.
IQL>from hotel.
IQL>where name=[shimoda_view_hotel].
IQL>
[[1,sel,p,840810120000,hotel.d..
          1='7368696d6f64615f766965775f686f74656c',*].

   [2.pro,t,1.2.6.7.ir42]]
[[1,get,i,ir42]]


  address                                                phone
  ------------------------------------------------------------
  633_ohara_kakizaki_shimoda_city                        (05582)2-6600
```

4. Demonstration of IRIS

The previous session showed you the IQL system, which is a combined system of RDB and map display functions. This session shows you IRIS, which combines Prolog and RDB. IRIS uses Prolog rules as deductive part and facts stored in Delta as data part. Thus IRIS can be regarded as a kind of deductive database system.

A relation, "neighbor", which stores neighborhood relations of resorts is used as the basic database in this example. However, any relations in Delta can be used as basis if we store appropriate set of rules in IRIS. The example is simmilar to the one in the scenario, but a little different.

IDB please.

```
idb1((neighbor(X,Y,DS,DR):-edb(neighbor(X,Y,DS,DR)))).
idb1((near(X,Y):-neighbor(X,Y,_1,_2))).
idb1((near(X,Y):-neighbor(Y,X,_1,_2))).
idb1((route(X,Y):-neighbor(X,Y,_1,_2))).
idb1((route(X,Y):-check,neighbor(X,Z,_1,_2),route(Z,Y)).
```

Now, please look at these rules in Prolog. The first rule (with edb) says relation "neighbor" is stored in Delta. If a resort X is a neighbor of another resort Y, then there is either "neighbor(X,Y)" or "neighbor(Y,X)" in Delta but not both to avoid redundancy. But the neighborhood is essentially a symmetric relation. Threfore, we define a kind of virtual relation "near" as a set of rules in Prolog. X is near Y if X is a neighbor of Y or Y is a neighbor of X. These two rules express this definition.

If a user asks about "near" relation in Prolog, then IRIS answers it using the definition of "near" and real relation "neighbor" in Delta.

---------- Query ----------

Thus, when a user want to know which resorts are near "shimoda', he can ask in Prolog as follows.

```
| ?-iris(idb1,near(shimoda,X)).
```

(First plan and Delta commands)

```
plan  :  [[neighbor,shimoda,X,_1_19,_2_19]]
optp  :  [[neighbor,shimoda,X,_1_19,_2_19]]
memo  :  pl([[neighbor,shimoda,X,_1_19,_2_19]],[[ir58,X]])

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 [[1,sel,p,840810120000,neighbor,d,1='7368696d6f6461',*],

    [2,pro,t,1,1,2,ir58]]
```

This is the first plan generated based on the first rule. This (opt) means the optimized plan. And, this (memo) indicates relations to be used later for optimization. Optimization does not change the plan in this example.

(Second plan and Delta commands)

```
plan  :  [[neighbor,X,shimoda,_1_19,_2_19]]
optp  :  [[neighbor,X,shimoda,_1_19,_2_19]]
memo  :  pl([[neighbor,X,shimoda,_1_19,_2_19]],[[ir59,X]])

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

 [[1,sel,p,840810120000,neighbor,d,2='7368696d6f6461',*],

    [2,pro,t,1,1,1,ir59]]
```

The second plan is generated according to the second rule.

```
[[1,unn,i,ir58,i,ir59,d,ir60]]
[[1,equ,i,ir60,i,ir58,d]]
equ_ans  :  false
[[1,get,i,ir60]]
```

This command (unn), union, combines the results of two plans.

This command (equ), is the equality check to avoid infinite loops.

Now, we get the result listed.

```
ir60('amagi          ').
ir60('inatori        ').
ir60('matsuzaki      ').
ir60('rendaiji       ').
ir60('yumigahama     ').
```

In this session, we have shown only a simple example. But IRIS can process rules that involve recursions and negative rules. With these functions, IRIS can process essentially any kind of rules in Prolog. Thus, IRIS is a step toward the development of knowledge base machine.

5. Questions and Answers

Now, we have shown you sample sessions with Delta. It's time for questions and answers.

.....

Thank you very much!

END of the DEMONSTRATIONS.

Appendix A

DATABASE SCHEMAS FOR THE DEMONSTRATION

(1) "resort" relation

name	Name of the resort
prefecture	Name of the prefecture in which the resort is located
distance	Distance from Tokyo to the resort (kilometers)
transportation	Transportation from Tokyo to the resort
fare	A single fare using the transportation (thousand yen)
souvenir1	A famous souvenir of the resort (in English)
souvenir2	A famous souvenir of the resort (in Japanese)
remarks	beauty spots, other souvenirs, etc.

(2) "recreation" relation

name	Name of the recreation
resort_name	Name of the resort where the recreation can do
fee	Fee for the recreation (thousand yen)

(3) "hotel" relation

name	Name of the hotel
resort_name	Name of the resort where the hotel is located
capacity	Capacity of the hotel
rate	Daily rate per room (thousand yen)
accommodations	Accommodations (e.g. single,bath,breakfast)
address	Address of the hotel
phone	Telephone number of the hotel

(4) "neighbor" relation

resort_name1	Name of the resort 1
resort_name2	Name of the resort 2 (farther than 1 from Tokyo)
distance	Distance between the resort 1 and 2
direction	Direction from the resort 1 to 2 (16 dir. e.g. sw, nne)

(5) "resort_map" relation

area_name	Name of district
map_data	Plotting data of district

(6) "location" relation

resort_name	Name of the resort
area_name	Name of district where the resort is located
symbol	Map symbol for the resort (including location)

Appendix B

"A Trip with Delta"

A famous database designer has come to Japan to participate in the International Conference on Fifth Generation Computer Systems, 1984. Using this opportunity, he plans to go sightseeing around Tokyo. However, he does not have enough information about resorts in Japan.

When he visits ICOT, he finds the relational database machine, Delta. Fortunately, it includes a database of Japanese resorts. He asks an ICOT researcher to use Delta to search for appropriate resorts. The researcher willingly accepts his request.

The researcher sits down in front of a video display terminal. That terminal is connected to a personal computer, and that computer is connected to Delta via a local area network. The researcher asks the designer, "how far are you willing to travel?" The designer says "I'd like to go there and come back in two days." The researcher replys "OK," while using keyboard to input the following sequence. As he does so, a map of central Japan appears on the video display.

IQL>*display.*
IQL>*select map_data.*
IQL>*from resort_map.*
IQL>*where area_name=[honshu].*

The map data is also stored in Delta.

"As you want to return within two days, we will search for resorts which are 100 to 150 kilometers distant from Tokyo", the researcher says. He then inputs the following sequence:

IQL>*locate.*
IQL>*select symbol.*
IQL>*from location.*
IQL>*where area_name=[honshu] and resort_name=next.*
IQL>*select name.*
IQL>*from resort.*
IQL>*where distance>[100] and distance<[150].*

Then, resorts satisfying the given conditions blink on the screen. "To which spot would you like to go?", asks the researcher. The designer points to a spot on the screen. "OK," the researcher responds, and inputs the following query:

IQL>*select name,distance,transportation,souvenir1.*
IQL>*from resort.*
IQL>*where distance>[100] and distance<[150] and prefecture=[shizuoka].*
IQL>*order by distance.*

The result table appears on the terminal screen. The designer crys "Oh! I have a weakness for dried fish!" The researcher, smiling, continues to type at the keyboard.

IQL>*display.*
IQL>*select map_data.*
IQL>*from resort_map.*
IQL>*where area_name=[izu].*

A new map appears in the screen. This detailed map contains a spot to which the designer points. The researcher continues to strike the keyboard.

IQL>*locate.*
IQL>*select symbol.*
IQL>*from location.*
IQL>*where area_name=[izu] and resort_name=next.*
IQL>*select name.*
IQL>*from resort.*
IQL>*where distance>[100] and distance<[150] and*
prefecture=[shizuoka] and souvenir1=[dried_fish].

The resort symbols start to blink. "These resorts are 100 to 150 kilometers from Tokyo and belong to Shizuoka prefecture," says the researcher. "You can buy good dried fish at any of these."

"Can we see what we can do there?" asks the designer. "Sure," says the researcher.

IQL>*select resort-name,recreation-name,recreation-fee,resort-fare.*
IQL>*from resort,recreation.*
IQL>*where resort-name=recreation-resort_name and*
resort-distance>[100] and resort-distance<[150] and
resort-prefecture=[shizuoka] and resort-souvenir1=[dried_fish].

"Good. I think I'll go to Shimoda," the designer says. "Then, we have to find a hotel for you to stay in," says the researcher, "do you have any requests concerning the hotel?" "Well, I'd prefer a rate of less than 15 thousand yen per night." "I see," the researcher replys, and enters this new information at the keyboard.

IQL>*select name,rate,capacity,accommodation.*
IQL>*from hotel.*
IQL>*where rate=<[15] and resort_name=[shimoda].*
IQL>*order by rate.*

A list of hotel names appears.

"I think Shimoda View Hotel is best," says the designer. "OK," says the researcher "let's get the address and telephone number."

IQL>*select name,address,phone.*
IQL>*from hotel.*
IQL>*where name=[shimoda_view_hotel].*

The designer notes the address and telephone number in his memorandum and says, "by the way, are there any good resorts I can visit when I come back to Tokyo?" "Well, we just happen to have a good system named IRIS for such questions. In this system, we can write our query using Prolog. The IQL processor is also written in Prolog, but it only converts IQL queries into Delta commands. We plan to use IRIS as a tool for investigating knowledge base systems. OK, let's have a look at resorts on the way back."

idb1((route(X,Y):-neighbor(X,Y))).
idb1((route(X,Y):-check,neighbor(X,Z),route(Z,Y))).
idb1((neighbor(X,Y):-edb(neighbor(X,Y)))).

"This is a Prolog program which describes the relationship between two resorts on one route. If we want to search for resorts on the route from Shimoda to Tokyo, we input the following Prolog query,

?- iris(idb1,route(X,shimoda)).

and resorts along the route are displayed."

"Thank you," says the designer, "I'll make my plans using this information." "You're welcome," says the researcher, "have a good time on your trip!"

■