DEPARTMENT OF COMMERCE (CA)
DATABASE MANAGEMENT SYSTEM (Semester-III)
II B.COM(CA) Sub Code-18BCA32C
UNIT - IV

Hierarchical Approach : IMS data structure - Physical Database, Database Description- Hierarchical sequence - External level of IMS : Logical Databases, the program communication block IMS Data manipulation : Defining the Program communication Block : DL / 1 Examples.

IMS data structure(Information Management System)

A physical database is an ordered set, the elements of which consist of all occurrences of one type of physical database record (PDBR).A PDBR occurrences in turn consists of a hierarchical arrangement of fixed-length segment occurrences; and a segment occurrence consists of a set of associated fixed-length field occurrences.

As an example we consider a PDB that contains information about the internal education system of a large industrial company. The hierarchical structure of this PDB-that is the PDBR type is shown here

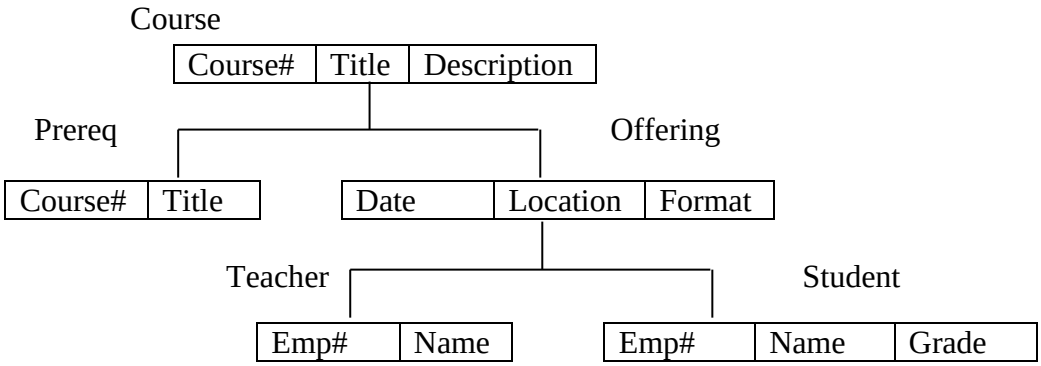


Fig: PDBR type for the education database.

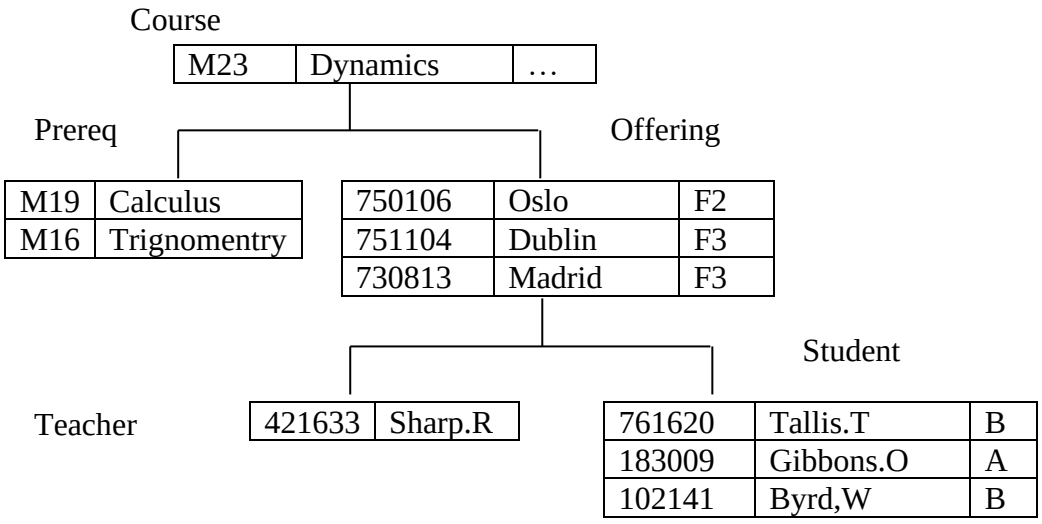
In this example we are assuming that the company maintains an education department whose function is to run a number of training courses. Each course is offered at a number of different locations within the company. The PDB contains details both of offerings already given and of offerings scheduled to be in the future,. The details are as follows:

- For each course: course number (unique), course title, course description, details of prerequisites courses if any, and details of all offerings.
- For each prerequisite course for a given course: course number and title.
- For each offering of a given course: date, location, format, details of all teachers and details of all students;
- For each teacher of a given offering: employee number and name
- For each student of a given offerings: (EMP_N), name and grade.

In the PDBR structure shown, we have five types of segments:

COURSE, PREREQ, OFFERING, TEACHER and STUDENT, each one consisting of the field types indicated.

COURSE is the root segment type and the others are department segment types. Each dependent has a parent for example the parent of TEACHER is OFFERING. Similarly each parent has at least one child, for example COURSE has two children. For one occurrence of any given segment type may be any number occurrences of each of its child segment types.



Sample PDBR Occurrence for the education database.

The database Description

Each physical database is defined together with its mapping to storage by a database description (DBD). The source form of the DBD is written using special System/370 Assembler language macro statements, once written the DBD is assembled and the object form is stored away in a system library, from which it may be extracted when required by the IMS control program. So the following is the DBD for the education database.

```
1 DBD       NAME=EDUCPDBD
2 SEGM      NAME=COURSE, BYTES=256
3 FIELD     NAME=(COURSE#, SEQ), BYTES=3,START=1
4 FIELD     NAME=TITLE, BYTES=33,START=4
5 FIELD     NAME=DESCRIPN, BYTES=220,START=37
6 SEGM      NAME=PREREQ, PARENT=COURSE, BYTES=36
7 FIELD     NAME=(COURSE#, SEQ), BYTES=3,START=1
8 FIELD     NAME=TITLE, BYTES=33,START=4
9 SEGM      NAME=OFFERING, PARENT=COURSE, BYTES=20
10 FIELD    NAME=(DATE, SEQ, M), BYTES=12,START1
11 FIELD    NAME=LOCATION, BYTES=12,START=19
12 FIELD    NAME=FORMAT, BYTES=2,START=19
```

```
13 SEGM     NAME=TEACHER,PARENT=OFFERING,BYTES=24
14 FIELD    NAME=(EMP#, SEQ), BYTES=6,START=7
15 FIELD    NAME=NAME, BYTES=18,START=7
16 SEGM     NAME=STUDENT,PARENT=OFFERING, BYTES=25
17 FIELD    NAME=(EMP#, SEQ), BYTES=18MSTART=7
18 FIELD    NAME=NAME, BYTES=18,START=7
19 FIELD    NAME=GRADE, BYTES=1,START=25
```

FIG: DBD for the education PDB.

Explanation

- Statement 1:Assigns the name EDUCPDBD ("education physical database description") to the DBD.All the names in IMS are limited to a maximum length of eight characters.

- Statement 2:Defines the root segment type with the name COURSE and has totally 256 bytes length.

- Statement 3-5:Defines the field types that go to make up COURSE. Each is given a name, a length in bytes, and a start position within the segment. The first field, COURSE# is defined to be the sequence field for the segment. So the PDBR occurrences will be sequenced in ascending course number order.

- Statement 6:Defines PREREQ as a 36-byte segment and is dependent on COURSE.

- Statements 7-8:Define the fields of PREREQ.

- Statement 9:Defines OFFERING as a child of COURSE.

- Statements 10-12:Define the fields of OFFERING.DATE are defined as the sequence field for OFFERING. The specification M (multiple) means that twin OFERING occurrences may contain the same date value.

- Statements 13-15:Define the TEACHER segment and its fields

- Statements 16-19:Define the STUDENT segment and its fields

The sequence of statements in the DBD is significant. Specifically SEGM statements must appear in the sequence that reflects the hierarchical structure also each SEGM statement must be immediately followed by the appropriate FIELD statements.

Hierarchical Sequence

The concept of hierarchical sequence within a database is a very important one in IMS.The definition for this is as follows:

For each segment occurrence, we define the "hierarchical sequence key value" to consist of the sequence field value for that segment, prefixed with the type code for that segment, prefixed with the hierarchical sequence key value of its parent, if any. For example, the hierarchical sequence key value for the STUDENT occurrence for "Byrd,W." is

1M2337308135102141

Here 1 is the type code for COURSE, M23 the course#, 3 is the type code of OFFERING, 730813 is the DATE of OFFERING, 5 is the type code of STUDENT, 102141 is the EMP# of STUDENT.

Then the hierarchical sequence for an IMS database is that sequence of segment occurrences defined by ascending values of the hierarchical sequence key. This notion is important in case of IMS databases because in IMS databases are stored in hierarchical sequence.

External Level OF IMS

Logical databases:

In architecture the user's external view was defined as subset of the corresponding physical database. A LDB (logical database) is an ordered set, the elements of which consist of all occurrences of one type of LDBR (logical database record).An LDBR type is a hierarchical arrangement of segment types, and is derived from the corresponding PDBR hierarchy in accordance with the following rules.

- Any segment type of the PDBR hierarchy together with all its dependents can be omitted from the LDBR hierarchy
- The fields of an LDBR segment type can be a subset of those of the corresponding PDBR segment type, and can be rearranged within that LDBR segment type.

Example:

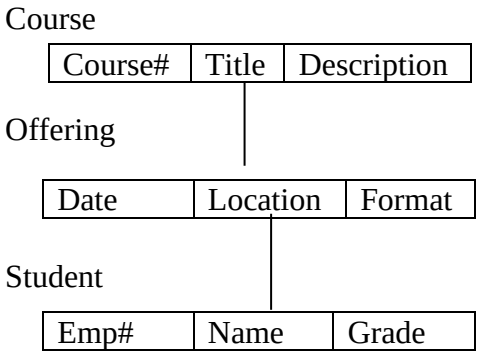


Fig: Sample LDBR type for the education database.

Sensitive Segments:

The segments, which are present in PDB and is included in LDB are said to be sensitive segments. In the above example COURSE, STUDENT, OFFERING are sensitive segments. The user of this LDB will not be aware of the existence of any other segments.
For example, the DL/I "get next" operation, which in general is used for sequential retrieval, will simply skip over any segments that are not sensitive for the user. If the user deletes a sensitive segment all children of that segment will be deleted regardless of

sensitiveness. So the user should not be given the authority to delete a segment, which allows the deletion of other hidden segments too.
Also sensitive-segment concept protects the user from modification like addition to the PDB unless it is proved that the addition of new segment may not affect any existing parent-child relationship.
Also sensitive-segment concept provides a degree of control over data security, is as much as users can be prevented from accessing particular segment types by the omission of those segments from the LDB.

Sensitive fields

Sensitive fields are those fields of the PDB that are included in the LDB.Every sensitive field must be controlled within a sensitive segment A given LDB may include or exclude any combination of fields from the PDB, in general except that if the program intends to insert new occurrences of a given segment type, then it must be "sensitive to" the sequence filed for that segment type.

Field sensitivity, like segment sensitivity, protects the user from certain types of growth in the database and provides a simple level of data security.

The program communication block (PCB)

Each LDB is defined by a PDB.The PCB includes the specification of the mapping between the LDB and the corresponding PDB.Like DBD (database description) a PCB is written using special system/370 assembler language macro statements. These statements constitute the "external DDL"for IMS.The set of all PCBs for a given user forms that user's program specification block (PSB); the object form of the PSB is stored in a system library, from which it may be extracted when required by the IMS control program.

Example:

```
1    PCB       TYPE=DB,DBNAME=EDUCPDBD,KEYLEN=15
2    SENSEG    NAME=COURSE, PROCOPT=G
3    SENSEG    NAME=OFFERING,PARENT=COURSE,PROCOPT=G
4    SENSEG    NAME=STUDENT,PARENT=OFFERING, PROCOPT=G
```

Explanation

- Statement 1:Specifies that this is a PCB database and named as EDUCPDBD, length of the key feedback area is 15 bytes.

  Key Feedback: When the user accesses an LDB, the corresponding PCB is held in storage and acts, as a communication area between the user's program and IMS.One of the fields in the PCB is the key feedback area. When the user retrieves a segment from the LDB, IMS not only fetches the requested segment but also places a "fully concatenated key" into the key feedback area.

  The fully concatenated key consists of the concatenation of the sequence field values of all segments in the hierarchical path from the root down to the retrieved segment.

Fetches the requested segment

For example;

Retrieve the STUDENT occurrence for Byrd.W.

IMS will place the value M23730813102141 in the key feedback area. The fully concatenated key of a segment is not quite the same as the "hierarchical sequence key" as this does not include segment type code information.

- Statement 2:Specifies the first sensitive segment in the LDB.The name of the sensitive segment must be same as the name assigned to the segment in the DBD.

  The PROCOPT (processing options") entry specifies the types of operation that the user will be permitted to perform on this segment. In this example the entry is G ("get") indicating retrieval only.
  Other options are I ("insert"), R ("replace") and D ("delete").
- Statement 3:Defines the next sensitive segments in the LDB.
- Statement 4:Defines the last sensitive segments. In our example statements 3 and 4 are very similar. The PROCOPT entry is the same for each of the three sensitive segments .In such a situation we may specify PROCOPT in the PCB statement instead of in each SENSEG statement.

  If PROCOPT=K is specified in the SENSEG statement for OFFERING, the user may largely ignore the presence of OFFERINGs in the hierarchy. The output for this modification is shown as follows.

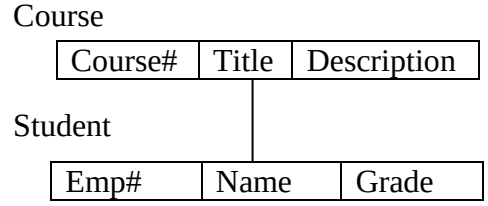


Fig: Effect of specifying PROCOPT=K for offering

The main difference is that when a STUDENT occurrence is retrieved, the fully concatenated key in the key feedback area will include the date value from the parent OFFERING.

The LDB shown in the example figure 1, is sensitive to all fields in segments COURSE, OFFERING and STUDENT of the underlying PDB.Suppose if we wish to exclude the LOCATION field of the OFFERING segment from the LDB while still remaining sensitive still all other fields as shown here:

```
SENFLD    NAME=FORMAT, START=1
SENFLD    NAME=DATE, START=1
```

These statements specify the fields to be included in the LDB segment and their start position within that segment. If no SENFLD statement is given for a particular SENSEG statement, then by default that segment is taken to be identical to the underlying PDB segment.

IMS Data Manipulation

Defining the Program Communication Block (PCB)

The IMS data manipulation language (DL/I) is invoked from the host language (PL/I) by means of ordinary subroutine calls. When an application program is operating on a particular logical database (LDB), the PCB for that LDB is kept in storage to serve as a communication area between the programs and IMS; infact when the program calls DL/I, it has to quote the storage address of the appropriate PCB to identify to DL/I which LDB it is to operate on.

PCB address is supplied to the program by IMS when the program is first entered. what actually happens is this.when a database application is to be run, IMS is given control first. IMS determines which PSB and DBD(s) are required, fetches them from their respective libraries and loads them into storage. IMS then fetches the application program and gives it control, passing it the PCB address as parameters.

In order for the application program to be able to access the information in the PCB for a particular LDB, it must contain a definition of that PCB.

```
DLITPLI:  PROCEDURE (COSPCB_ADDR) OPTIONS (MAIN);
          .
          .
          .
Declare   1   COSPCB       BASED(COSPCB_ADDR),
          2   DBDNAME      CHARACTER(8),
          2   SEGLEVEL     CHARACTER(2),
          2   STATUS       CHARACTER(2),
          2   PROCOPT      CHARACTER(4),
          2   RESERVED     FIXED BINARY(31),
          2   SEGNAME      CHARACTER(8),
          2   KEYFBLEN     FIXED BINARY(31),
          2   #SENSEGS     FIXED BINARY(31),
          2   KEYFBAREA    CHARACTER(15);
```

Explanation:

The procedure statement (labeled DLITPLI) is the program entry point. the expression in parentheses following the keyword PROCEDURE represents the parameters to be passed to the program by IMS, it consists of the pointer giving the address of the PCB.

The field DBDNAME contains the name of the underlying DBD throughout the execution of the program.

The SEGLEVEL field is set after the DL/I operation to contain the segment level number of the segment just accessed.
The STATUS field is the most important field in the PCB. After each DL/I call, the two character value is placed in this field to indicate the success or otherwise of the requested operation. A blank value indicates that the operation was completed satisfactorily, any other value represents an exceptional or error condition.

The PROCOPT field contains the PROCOPT value as specified in the PCB statement when the PCB was originally defined.

The SEGNAME field contains the name if the segment last accessed.

The KEYFBLEN field contains the length of the fully concatenated key.

The #SENSEGS field contains a count of the number of sensitive segments.

The field KEYFBAREA is the key feedback area contains the fully concatenated key.

DL/I Examples

Get Unique (GU)	Direct retrieval
Get next (GN)	Sequential retrieval
Get next with parent (GNP)	Sequential retrieval under current parent
Get hold (GHU), (GHN),(GHNP)	Allows subsequent DLET/REPL
Insert (ISRT)	Add new segment occurrence
Delete (DLET)	Delete existing segment occurrence
Replace (REPL)	Replace existing segment occurrence

Tab: DL/I

OPERATIONS

Direct retrieval:

Get the first OFFERING occurrence where the location is Stockholm.

```
GU    COURSE
      OFFERING (LOCATION ='STOCKHOLM')
```

Sequential retrieval with an SSA:

Get all STUDENT occurrences in the LDB, starting with the first student for the first offering in Stockholm.

```
      GU    COURSE
            OFFERING (LOCATION='STOCKHOLM')
            STUDENT
NS    GN    STUDENT
      GOTO  NS
```

Sequential retrieval with an SSA within a parent:

Get all students for the offering on 13 august 1973 of course M23.

```
GU    COURSE (COURSE#='M23')
      OFFERING (DATE='730813')
```

```
NP      GNP   STUDENT
        GOTO  NP
```

Segment occurrence insertion:

Add a new segment occurrence for the offering on 13 august 1973 of course M23.

```
ISRT   COURSE (COURSE#='M23')
       OFFERING (DATE='730813')
       STUDENT
```

Segment deletion:

Delete the offering of course M23 on aug 1973.

```
GHU    COURSE (COURSE# = 'M23')
       OFFERING (DATE='730813')
DLET
```

Segment replacement:

Change the location of the 13 Aug 1973 offering of course M23 to Helsinki.

```
GHU    COURSE (COUSE# ='M23')
       OFFERING (DATE='730813')
REPL
```

BOOKS FOR REFERENCE:

1. Database systems concepts by Abraham Silberschatz, Henry F. Korth.
2. An Introduction to Database System – C. Dsai.
3. An introduction to Database Systems (Seventh Edition) – C.J. Date

Prepared by Dr.N.SHANMUGAVADIVU