

**RASHTRIYA SANSKRIT SANSTHAN, LUCKNOW CAMPUS**

**SHASTRI – 1<sup>ST</sup> YEAR (2<sup>ND</sup> SEM)**

**SUBJECT- COMPUTER (PAPER-07)**

**NOTES**



**PROVIDED BY  
SHATAKSHI MISHRA  
GUEST FACULTY- COMPUTER  
RASHTRIYA SANSKRIT SANSTHAN  
LUCKNOW CAMPUS**

## **FoxPro Files**

1. **TABLE** – किसी भी डेटाबेस में table सबसे महत्वपूर्ण यूनिट होती है | Table के द्वारा ही डेटाबेस स्ट्रक्चर डिफाइन किया जाता है | Table सम्बंधित रिकार्ड्स या data elements का वह समूह है, जो rows और कोलम्स के रूप में व्यवस्थित होता है |

### **Table by Wizard:**

- I. File menu → NEW → TABLE → Wizard
- II. Select Table ---→ select fields
- III. Modify field settings
- IV. Index the table
- V. Finish

### **Table by Designer Window :**

- I. File menu → NEW → TABLE → New File
- II. Save the table with a Name
- III. Add the required fields
- IV. Press OK and add the records

2. **QUERY** – डेटाबेस से एक फिक्स फॉर्मेट और सिंटेक्स में इनफार्मेशन अथवा सूचना प्राप्त (retrieve) करने के लिए दिया गया कमांड query कहलाता है| query की सहायता से अलग अलग टास्क परफॉर्म किये जाते हैं एवं इसका प्राइमरी पर्पस किसी क्राइटेरिया के बेस पर डेटाबेस से किसी विशिष्ट data को फिल्टर करके सूचना प्रदाम करना होता है |  
क्वेरीज, data को calculate या summarize करने के साथ साथ data management कार्यो को भी ऑटोमेट करती हैं|

### **Table by Wizard:**

- I. File menu → NEW → QUERY → Wizard
- II. Select Table ---→ select field
- III. Filter Records: इस स्टेप में अपनी query की condition लिखें| यहाँ आप अधिकतम दो condition दे सकते हैं|
- IV. Sort Records: इस स्टेप में आप किसी भी field को आधार बना कर अपने रिकार्ड्स को आरोही अथवा अवरोही क्रम में व्यवस्थित कर सकते हैं|
- V. Finish

### **Table by Designer Window :**

- I. File menu → NEW → QUERY → New File
- II. Add table की विंडो में Other ऑप्शन से table सेलेक्ट करें।
- III. अब query डिज़ाइनर विंडो में कार्य करना शुरू करें।
- IV. पहले field टैब से अपनी table के जिन फ़िल्ड्स को आप डिस्प्ले कवना चाहते हैं उनका सिलेक्शन करें।
- V. उसके बाद Filter टैब में जाकर अपनी कंडीशन लिखें।
- VI. आउटपुट प्राप्त करने के लिए Ctrl+Q अथवा Query menu में जाकर query को run करवाएं।

3. **FORM** – फॉर्म visual foxpro में एक ऐसी फाइल है जो डेटाबेस एप्लीकेशन के लिए यूजर इंटरफ़ेस बनाने में सहायता करती है। फॉर्म किसी data स्रोत जैसे किसी table या query से सीधे कनेक्ट होता है, तथा उस स्रोत से data स्टोर करने, एक्सीक्यूट करने अथवा डिस्प्ले करने के लिए प्रयोग में लाया जाता है।

फॉर्म का प्रयोग यूजर से data इनपुट करवाने के लिए किया जाता है। फॉर्म में अधिक संख्या में data को स्टोर करके रखा जा सकता है। इस काम को करने के लिए फॉर्म में विभिन्न प्रकार के ऑप्शन होते हैं जैसे new record add करना, रिकॉर्ड को delete करना, रिकॉर्ड को edit करना, रिकॉर्ड्स में formatting तथा मूव करना आदि।

4. **REPORT** – डेटाबेस में रिपोर्ट एक ऐसी फाइल है जिसे data को संगठित तरीके से डिस्प्ले और प्रिंट करने के लिए प्रयोग किया जाता है। foxpro में table से रिपोर्ट बनाने की सुविधा है, ये डेटाबेस का सारांश होती है। रिपोर्ट को 3 section में बांटा जा सकता है। इसमें प्रत्येक section का अपना कार्य और व्यवहार होता है। ये 3 section निम्न हैं-

- HEADER
- BODY
- FOOTER

**HEADER** : header रिपोर्ट के सबसे ऊपर का हिस्सा होता है। इसमें रिपोर्ट की हैडिंग देते हैं, जो यह निर्देशित करता है की रिपोर्ट किस चीज़ से सम्बंधित है। इसमें table के फ़िल्ड्स के label भी दिए जा सकते हैं।

**BODY** : यह रिपोर्ट का main एरिया होता है, इसमें table के field को सेट किया जाता है।

**FOOTER** : यह रिपोर्ट का सबसे निचला हिस्सा होता है, जिसमें रिपोर्ट का सारांश होता है।

5. **MAIL MERGE** – मेल merge एक ऐसी मास-मेलिंग सुविधा है जिसके द्वारा आप एक ही पत्र अनेक व्यक्तियों को भेज सकते हैं, ये पत्र कोई प्रवेश पत्र, निमंत्रण पत्र, ऑफिस लैटर आदि हो सकता है।

Foxpro में यह कार्य आसानी से किया जा सकता है, जिसके लिए पहले हम पत्र जिन व्यक्तियों को भेजा जाना है उनके डेटेल्स की एक table तैयार करते हैं और फिर ही भेजे जाने वाले पत्र को उसके साथ सम्मिलित कर देते हैं। इसमें डॉ तरह की फाइल्स बनाई जाती हैं - data source व main document |

**MAIL MERGE, TOOL menu के WIZARD टैब में होता है।**

- **Data Source** : data सोर्स सूचना का एक व्यवस्थित एप्लीकेशन संग्रह जो visual foxpro में table के रूप में स्टोर होता है। यह दूसरे एप्लीकेशन का भी data प्रयोग करता है जैसे MS ACCESS या MS EXCE का। ऐसे data सोर्सज में रिकार्ड्स और फ़िल्ड्स होते हैं अर्थात उन व्यक्तियों के नाम, address तथा अन्य जानकारियाँ होती हैं जिन्हें पत्र भेजा जाना है।
- **Main Document** : मेन डॉक्यूमेंट में भेजे जाने वाला पत्र होता है, जैसे कोई greeting कार्ड, या बर्थडे इनविटेशन आदि |

## What is sorting?

- Sorting वह प्रक्रिया है जिसके द्वारा हम डेटा को एक logical order में arrange (क्रमबद्ध) करते हैं | यह लॉजिकल ऑर्डर ascending order (आरोही) भी हो सकता है या descending order (अवरोही) भी हो सकता है | ascending का अर्थ होता है कि बढ़ते क्रम में और descending का अर्थ होता है घटते क्रम में.
- Sorting एक important operation है, जो alphabetical, numerical अथवा chronological किसी भी order में किया जा सकता है |
- sorting प्रायः एक list of elements पर perform की जाती है , जो सर्चिंग प्रक्रिया को काफी कुशलता व शीघ्रता से करने में सहायता करती है |
- Types of sorting (सॉर्टिंग के प्रकार) : यह दो प्रकार की होती है:-
  - 1:- Internal सॉर्टिंग
  - 2:- External सॉर्टिंग
- ऐसे list की sorting को internal sorting कहते हैं, जिसमें sort किये जाने वाला सभी डेटा main memory में ही रहता है |
- इसके विपरीत जब sort किये जाने वाला डेटा इतना ज्यादा होता है कि वह main memory में नहीं आ पाता तो इसे secondary memory में रखा जाता है, प्रकार की sorting external sorting कहते हैं|

## SORT command

SORT command के द्वारा किसी भी टेबल के डाटा को सॉर्ट किया जा सकता है , विमुअल फॉक्सप्रो सॉर्टेड डाटा को एक नई फाइल में स्टोर करता है | जिससे ओरिजिनल फाइल स्ट्रक्चर में कोई बदलाव नहीं होता व डाटा सुरक्षित रहता है |

Syntax: **SORT TO <new file> ON <field /A/D/C>, <fields....>**

**Note:** यदि आप किसी भी ऑप्शन को सेलेक्ट नहीं करते हैं तो फॉक्सप्रो बाई डिफॉल्ट असेंडिंग ऑप्शन में डाटा को सॉर्ट करता है |

जहां-

**A:** डाटा को आरोही क्रम में व्यवस्थित करता है |

**D:** डाटा को अवरोही क्रम में व्यवस्थित करता है |

**C :** ये option cases का ध्यान रखता है , लिस्ट में से कैपिटल लेटर पहले व स्माल लेटर बाद में आते हैं |

## Example :

```
USE EMPLOYEE // एम्प्लोयी टेबल ओपन करने के लिए
SORT TO NSORT ON NAME // NSORT नाम की नई फाइल में NAME के आधार पर डाटा सॉर्टिंग
```

- रिकार्ड्स सॉर्ट होकर स्टेटस बार में मेसेज के रूप में प्रदर्शित होते हैं |

**2 या 5 record sorted.**

- sorted रिकार्ड्स देखने के लिए BROWSE के साथ नई फाइल का नाम इंटर करें |

```
USE NSORT // नई फाइल जहाँ डाटा सॉर्ट किया गया है|
```

```
BROWSE // इसके द्वारा आप डाटा तो table के रूप में देख सकेंगे
```

- VISUAL FOXPRO में फाइल अपने आप सॉर्ट नहीं की जा सकती ,हर बार SORT ऑपरेशन क्रियान्वित किया जाता है, हर बार एक नई FILE बनानी होती है |
- SORT करने के लिए हमेशा ओरिजिनल फाइल की एक डुप्लीकेट फाइल बनती है , VIEW ये PRINT करने के लिए आपको इसी को प्रयोग में लाना पड़ता है |

### • **CONDITION के आधार पर -**

निम्न उदाहरण में डाटा अवरोही क्रम में डिस्प्ले होगा |

```
USE EMPLOYEE // एम्प्लोयी टेबल ओपन करने के लिए
SORT TO TEMP ON NAME / D //NAME डाटा यहाँ घटते हुए क्रम में प्रदर्शित होगा
USE TEMP
LIST
```

### • **USING FOR clause for SORT selected records –**

यदि आप चाहें तो कुछ सिलेक्टेड रिकार्ड्स को ही SORTED RECORDS की फाइल में शामिल कर सकते हैं | जिसके लिए आपको उचित scope व condition कमांड्स के साथ ही देनी होती है | SORT कमांड को FOR scope के साथ निम्न तरीके से प्रयोग में लाते हैं -

```
Syntax: SORT TO <new file> ON <field_list> FOR <logical_exp>
```

जहां-

**field\_list** : जिन फ़िल्ड्स के अधर पर डाटा सॉर्ट होना है।

**Logical\_exp** : वो condition जिनके आधार पर सॉर्टिंग होनी है ।

**Example:** दिए गए कोड के अनुसार EMPLOYEE फाइल से NAME फ़िल्ड के अनुसार सिर्फ उन्ही एम्प्लोयेस का डाटा सॉर्ट होगा जिनका SALES \_ID S50 है -

```
USE EMPLOYEE
SORT TO TEMPFILE ON NAME FOR SALES_ID = "S50"
USE TEMPFILE
LIST
```

- **Sorting data on various fields -**

यदि किसी परिस्थिति में आप एक से अधिक फ़िल्ड्स के आधार पर डाटा को सॉर्ट करन चाहते हैं, तो ये ऑप्शन आपकी सहायता करता है। यहाँ आप एक से अधिक फ़िल्ड्स को comma के माध्यम से अलग करके अपने कोड को एक्सीक्यूट कर सकते है ।

**Syntax:** SORT TO <new file> ON field1 [/A /C /D], field2 [/A /C /D]... last\_field

**Example:** USE EMPLOYEE

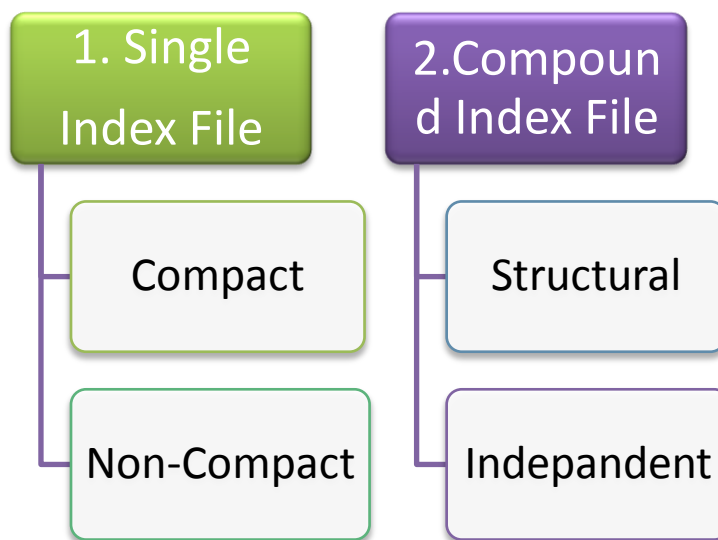
```
SORT TO TESTFILE ON NAME, DOJ
```

```
USE TESTFILE
```

```
BROWSE
```

# What is indexing?

- इंडेक्सिंग एक तरीका है जो डाटाबेस की table में डेटा पुनर्प्राप्ति की गति को सुधारने के लिए उपयोग किया जाता है।
- किसी table में एक या एक से अधिक स्तंभों का उपयोग करके एक index बनाया जा सकता है और index एक अलग फ़ाइल में store रहता है।
- इस फ़ाइल में table में उनकी भौतिक स्थिति के साथ rows का logical order होता है।
- किसी index फ़ाइल द्वारा आवश्यक स्थान आमतौर पर table को store करने के लिए आवश्यक स्थान से कम होता है।
- इंडेक्सिंग करते समय ओरिजिनल फाइल नहीं बदलती है।
- इंडेक्सिंग एक अलग खंड है जो ये बताता है की विसुअल फोकस्परो की फाइल में डेटा कहाँ स्थित है। अतः ये एक अलग फाइल होती है जिसमे रिकार्ड्स की स्थिति से सम्बंधित डेटा मूल फाइल में ही होता है ।
- विसुअल फोकस्परो में index file दो प्रकार की होती है -



## • Single Index File:-

ये फाइलें .IDX extension के साथ डिस्क पर सेव होती हैं | इन फाइलों के द्वारा केवल एक ही तरह के index को मैनेज किया जा सकता था | ये फाइलें मैनुअली अपडेट की जाती हैं | इन्हें आटोमेटिक अपडेट करने के लिए इन्हें एक्टिव या ओपन रखना आवश्यक होता है | ये फाइलें भी दो प्रकार की होती है -

Compact Index File	Non-Compact Index File
ये फाइलें एक्सेस होने में तीव्र व छोटे आकार की होने के कारण डिस्क में कम स्थान लेती हैं , जिससे ये अधिक प्रभावी होती हैं	
Single Entry Index बनाने के लिए Syntax होता है - USE <table_name> INDEX ON <expression> TO <.idx file> COMPACT	Single Entry Index बनाने के लिए Syntax होता है - USE <table_name> INDEX ON <expression> TO <.idx file>

### • Compound Index File:-

ये फाइलें .CDX extension के साथ डिस्क पर सेव होती हैं | IDX फाइलों से अलग ये फाइलें कई तरह के index को एक ही फाइल में सम्हाल सकती हैं, जिन्हें tags के नाम से जाना जाता है |

Structural compound index (.cdx) files	Independent/Nonstructural compound index (.cdx) files
<ul style="list-style-type: none"> <li>✓ Opens and closes with the table automatically.</li> <li>✓ Uses same base name as the table file name.</li> </ul>	<ul style="list-style-type: none"> <li>✓ Must be opened explicitly.</li> <li>✓ Uses a name different from the base table name and is defined by the user.</li> </ul>
<ul style="list-style-type: none"> <li>✓ Contains Multiple index keys.</li> </ul>	<ul style="list-style-type: none"> <li>✓ Contains Multiple index keys.</li> </ul>
<ul style="list-style-type: none"> <li>✓ Syntax: USE &lt;table_name&gt; INDEX ON &lt;exp&gt; TAG &lt;tag_name&gt;</li> </ul>	<ul style="list-style-type: none"> <li>✓ Syntax: USE &lt;table_name&gt; INDEX ON &lt;exp&gt; TAG &lt;tag_name&gt; OF &lt;.CDX file&gt;</li> </ul>
<ul style="list-style-type: none"> <li>✓ प्रायः स्थितियों में सभी index जब भी आप कोई फाइल अपडेट करते हैं, स्वतः ही अपडेट हो जाते हैं।</li> </ul>	<ul style="list-style-type: none"> <li>✓ इस तरह की फाइल को मैनुअली अपडेट करने की आवश्यकता रहती है  </li> </ul>
<ul style="list-style-type: none"> <li>✓ विसुअल फोक्सप्रो ओपन पर जैसे ही structural कंपाउंड फाइल को देखता है और बिलकुल उसी के नाम वाली .CDX फाइल को ढूँढता है,जैसे ही वो मिल जाती है उसे ओपन कर दिया जाता है  </li> </ul>	<ul style="list-style-type: none"> <li>✓ ये table फाइल ओपन होने पर स्वतः ही open नहीं होती हैं  </li> </ul>
<ul style="list-style-type: none"> <li>✓ maximum प्रयोग वाले tags इस फाइल में रखे जाते हैं  </li> </ul>	<ul style="list-style-type: none"> <li>✓ इस फाइल में उन tags को स्थान दिया गया है जिनका प्रयोग बहुत ही कम होता है अथवा जो महत्वहीन हों और structural compound फाइल में अधिक संख्या होकर updation में अधिक समय ले रहे हों।</li> </ul>

## • Deleting tags:-

- ✓ Compound Index File में से tags को delete करने के लिए DELETE TAGS कमांड का प्रयोग किया जाता है-

```
DELETE TAG <tag_name> OF <.CDX index_name>
```

नोट : यदि आप किसी नॉन-स्ट्रक्चरल फाइल से tags को डिलीट कर रहे हैं तो उसे ओपन करना अतिआवश्यक है।

- ✓ एक से अधिक tags डिलीट करने के लिए -

```
DELETE TAG <tag_name1> OF [<.CDX file1>], [<tag_name2>] [OF <.CDX file2>]
```

- ✓ किसी फाइल से सभी tags एकसाथ डिलीट करने के लिए-

```
DELETE TAG ALL < OF .CDX file>
```

## • Creating Index:-

INDEX कमांड का प्रयोग करके करंट फाइल के लिए INDEX बनाया जा सकता है।

### Syntax-

```
INDEX ON <field_expr> TO <.IDX file>  
[ TAG <tag_name> [OF <CDXfile>]  
FOR <condition>]  
[COMPACT] [ASCENDING / DESCENDING]  
[UNIQUE] [ADDITIVE]
```

- ✓ उक्त कमांड्स आपकी currenly ओपन फाइल के रिकार्ड्स को **field expression** पर आधारित क्रम में बदल देता है पर table के वास्तविक रिकार्ड्स की व्यवस्था में कोई परिवर्तन नहीं करता है ।
- ✓ **FOR** क्लॉज़ जोकि ऑप्शनल है, index को सिर्फ वही record शामिल करने देता है जिसके लिए लॉजिकल वैल्यू true है ।

COMPACT ऑप्शन भी ऑप्शनल होता है जो compact IDX फाइल बनाने की सहायता देता है। ये index फाइल्स को कॉम्प्रेस करके size में छोटा करता है और अधिक प्रभावी बनता है ।

- ✓ **ASCENDING / DESCENDING** फोकस्प्री को यह बताता है की फाइल को किसी विशेष field के आरोही व अवरोही क्रम की वैल्यू के अनुसार ही index करना है।  
वैसे तो आरोही क्रम को डिफॉल्ट माना गया है और IDX फाइल बनाते समय DESCENDING order का प्रयोग नहीं किया जा सकता है।
- ✓ **UNIQUE** यह ऑप्शन डुप्लीकेट record को हटाकर सिर्फ उसी वैल्यू को रखता है जो field expression के लिए दी गई वैल्यू के साथ पहली बार आता है।
- ✓ **ADDITIVE** जब भी आप कोई नई index file बनाते हैं तो यह पहले से ओपन फाइल्स को खुला ही रहने देता है।

#### • **Various Indexing Commands:-**

- ✓ **SET INDEX-** कमांड का प्रयोग करके आप किसी भी समय किसी भी index को किसी भी समय एक्टिव कर सकते हैं।
- ✓ **SET ORDER-** किसी भी index फाइल को activate करने के लिए इसका प्रयोग किया जाता है। आप इसके साथ किसी digit, tag name अथवा field के आधार पर प्रयोग किया जा सकता है।
- ✓ **REINDEX-** यह कमांड दुबारा से index फाइल को ओपन करता है व उसपे परफॉर्म होने वाले परिवर्तन को करता है तथा यदि फाइलें पहले ठीक तरह से close नहीं हुई थी तो उन्हें भी बंद करता है। ऐसी आवश्यकता पड़ती रहती है जैसे कभी आपने बंद फाइलों में कुछ अपडेट कर दिया पर वो परिवर्तन वहां हुआ नहीं तो reindexing करते हैं।
- ✓ **CLOSE INDEX-** आपका कार्य समाप्त होने पर इस कमांड का प्रयोग कर आप index files को close भी कर सकते हैं ।

#### • **SORTING v/s INDEXING**

- ✓ INDEX कमांड, SORT कमांड की अपेक्षा तेज़ काम करता है।
- ✓ INDEX FILE, SORT FILE की अपेक्षा डिस्क पर कम जगह लेता है ।
- ✓ INDEX कमांड नए record जोड़ने व पहले से उपस्थित रिकार्ड्स की मूल संरचना बनाये रखने में सहायक होता है ।

- **Query with FIND command**

क्योंकि FIND डाटा field की character string पे काम करता है, अतः यदि table को उस field पर index किया जाए जहाँ वह सूचना है जिसे ढूँढना है, तो FIND कमांड की सहायता से यह कार्य बहुत जल्दी हो जायेगा।

FIND सर्च को index किये हुए table के पहले record से तब तक सर्च करता है जबतक वो specified value न मिल जाए या वो table के end तक न आ जाये।

FIND का प्रयोग करने से पहले ये ध्यान अवश्य दें की index file पहले से open होनी चाहिए।

**Syntax: FIND CHARACTER STRING** //यहाँ string quotes(" ") के साथ नहीं लिखी जाएगी।

**EXAMPLE:** यदि आपको RAM नाम के छात्र को Student table से ढूँढना है तो-

**USE STUDENT INDEX NAMES**  
**FIND RAM**

//सर्च की हुई string की record संख्या प्रदर्शित हो जायेगी, और सही output न मिलने की स्थिति में पॉइंटर table के अंत में पहुच जाता है और STATUS BAR में EOF (END OF FILE) दिखेगा।

FIND कमांड condition से मिलने वाली पहली string को सर्च करके देता है, यदि आप ये देखना चचते हैं कि सर्च string से मिलने वाली और भी वैल्यूज table में हो सकती हैं तो WHILE क्लॉज़ का प्रयोग LIST कमांड के साथ करें।

**EXAMPLE:** यदि आपको RAM नाम के छात्र को Student table से ढूँढना है तो-

**USE STUDENT INDEX NAMES**  
**FIND "RAM"**  
**DISPLAY**  
**LIST WHILE NAME = "RAM"**

// NAMEfield में LIST WHILE सारे रिकार्ड्स में "RAM" को प्रदर्शित करता है क्योंकि सारे रिकार्ड्स "RAM" के साथ एक के बाद एक index फाइल में स्थिर होते हैं ।

- **Query with SEEK command**

SEEK कमांड में index की हुई table से वो पहले record ढूंढ सकते हैं जो आपकी दी हुई condition के expression से मैच करता हो | एक expression, character string या कोई mathematical value कुछ भी हो सकती है।

seek कमांड में दी गई गई वैल्यू को double quotes (“ ”) में बंद करते हैं |

**Syntax: SEEK expression** //यहाँ expression को (“ ”) के साथ लिखते हैं।

SEEK कमांड के द्वारा किसी वेरिएबल में स्टोर वैल्यू को index table के expression में दी गई वैल्यू से मैच करा के सर्च किया जाता है |

**EXAMPLE:** किसी इंडेक्स्ड फाइल से कोई वैल्यू ढूढने के लिए -

```
USE STUDENT INDEX NAMES
STORE "RAM KUMAR" TO XNAME
SEEK XNAME
```

//SEEK कमांड एक्टिव index फाइल में वो record सर्च करता है जिससे expression की वैल्यू “RAM KUMAR” से मैच होता है | फोक्सप्रो record को दिस्पले नहीं करता है, बस पॉइंटर को उस record पर पॉइंट करवाता रहता है |

// RECORD को DISPLAY कमांड की हेल्प से आप देख सकते हैं |

- **FIND v/s SEEK Command**

जैसा की आपने ऊपर दिए हुए उदाहरण की सहायता से देखा की SEEK कमांड वेरिएबल में स्टोर वैल्यू को सर्च करता है जबकि FIND कमांड अपने साथ ही लिखी वैल्यू को सर्च करेगा |

FIND XNAME पूरी table में XNAME शब्द को सर्च करेगा जबकि SEEK XNAME, XNAME में स्टोर वैल्यू “RAM KUMAR” को table में सर्च करेगा |