Toppersnotes
Unleash the topper in you

# PROGRAMMER

## Rajasthan Public Service Commission

**Volume 3**

कंप्यूटर साइंस

SIERRA INNOVATIONS

# Index

# Database Management System

## An overview of the Database management

- डाटाबेस, सूचनाओं (या डाटा) का एक ऐसा व्यवस्थित संग्रह (Organized Collection) होता है, जिससे हम किसी भी सूचना को सरलता से प्राप्त कर सकते हैं।
- डाटाबेस व्यवस्थित इसलिए होता है, क्योंकि इसमें किसी भी डाटा या सूचना को एक निश्चित स्थान पर पहले से तय किए हुए रूप में रखा जाता है, ताकि कभी भी आवश्यकता पड़ने पर उसे आसानी से ढूँढकर देखा जा सके।

**व्यवस्थित डाटाबेस में हमें निम्नलिखित कार्य की सुविधा होती है -**

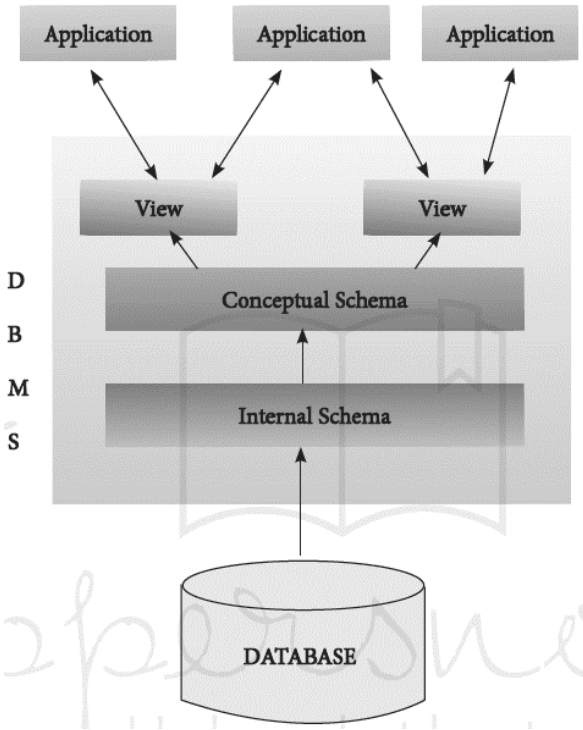
- आवश्यक सूचना को निकालना सूचनाओं के अनुसार उचित कार्यवाही करना या निर्णय लेना।
- सूचनाओं को नई आवश्यकताओं के अनुसार फिर से व्यवस्थित करना।
- सूचनाओं के आधार पर रिपोर्ट आदि बनाना तथा नई सूचनाएँ निकालना।
- एक डाटाबेस, नामों की सूची की एक फाइल के रूप में आसान भी हो सकता है और डाटा की बहुत सी-फाइलों के समूह के रूप में कठिन भी हो सकता है।

**डाटा (Data) -** किसी वस्तु, व्यक्ति या समूह के बारे में किसी तथ्य अथवा जानकारी को डाटा (Data) कहा जाता है। किसी व्यक्ति का नाम, किसी वस्तु का वजन तथा मूल्य, किसी कक्षा के विद्यार्थियों की उम्र आदि ये सभी डाटा के उदाहरण हैं।

**सूचना (Information)** - जब किसी डाटा को सार्थक तथा उपयोगी बनाने के लिए संसाधित, व्यवस्थित, संरचित किया जाता है, तो उसे हम सूचना कहते हैं।
उदाहरण के लिए एक कक्षा का औसत स्कोर एक सूचना है, जोकि उस कक्षा के विद्यार्थियों के स्कोर से निकाला जा सकता है। संक्षेप में, डाटा डाटाबेस में स्टोर मूल्यों को सन्दर्भित (Refer) करता है, जबकि सूचना उन मूल्यों से निकाले गए निष्कर्ष या अर्थ को सन्दर्भित करती है।

**डाटाबेस एब्सट्रेक्शन (Database Abstraction) -** डाटाबेस सिस्टम उपयोगकर्ता कों को केवल उनकी आवश्यकतानुसार डाटा उपलब्ध करवाता है और मेमोरी में कैसे डाटा संग्रहित और प्रबंधित होता है कि जानकारी छुपाता है। यह अवधारणा/प्रकिया डाटाबेस एब्सट्रेक्शन कहलाती है।

**डाटाबेस मैनेजमेंट सिस्टम आर्किटेक्चर (Database Management System Architecture)**
डाटाबेस मैनेजमेंट सिस्टम त्रि - स्तरीय (three tier) स्कीमा आर्किटेक्चर (Schema architecture) को निरूपित (describe) करता है। इसमें तीन सतह/स्तर (Levels) होते हैं। ये निम्नलिखित हैं -

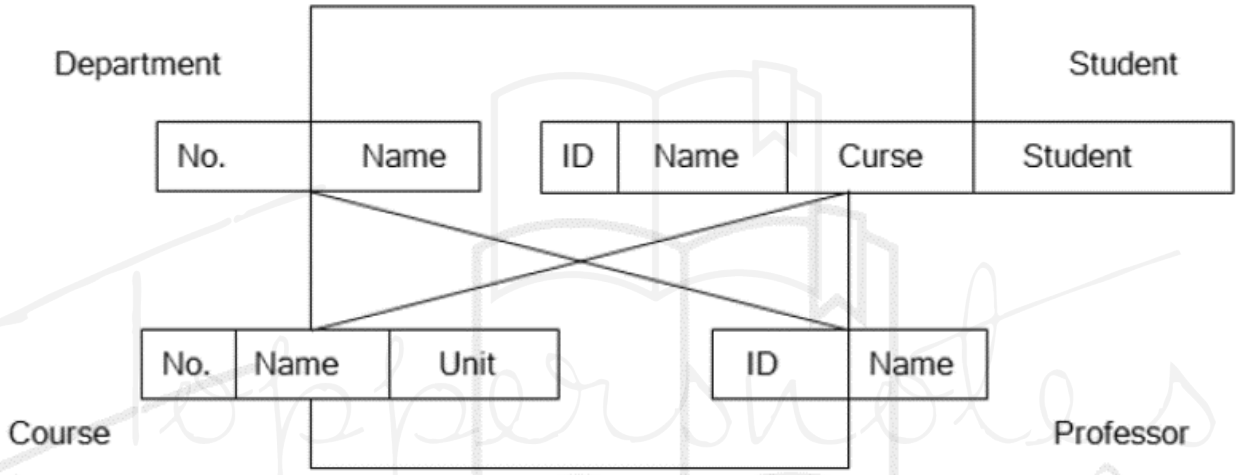


1. **Physical Level अथवा Internal Level -** यह निरूपित करता है कि डाटाबेस में डाटा कैसे स्टोर होता है ? यह abstraction का सबसे निचला सतह/स्तर (lowest level) है।
2. **Conceptual अथवा Logical Level -** Conceptual abstraction का अगला ऊँचा लेवल (next higher level) है। यह सतह/स्तर (level) निरूपित करता है कि डाटाबेस में क्या डाटा स्टोर रहता है और उनके मध्य क्या सम्बन्ध (relationship) है ? यह डाटाबेस प्रबन्धक (Database administrator) का कार्य होता है।
3. **External अथवा View Level -** ज्यादातर डाटाबेस उपयोगकर्ता (user) पूरे डाटाबेस का उपयोग नहीं करते हैं लेकिन उसका कुछ भाग ही पढ़ते हैं। इसलिए वे उस भाग के view level को उपलब्ध कराते हैं। यह abstraction का सबसे उच्चतम सतह/स्तर (highest level) है। यह किसी विशिष्ट ग्रुप के उपयोगकर्ता (user) के लिए डाटाबेस के एक भाग को निरूपित करता है।

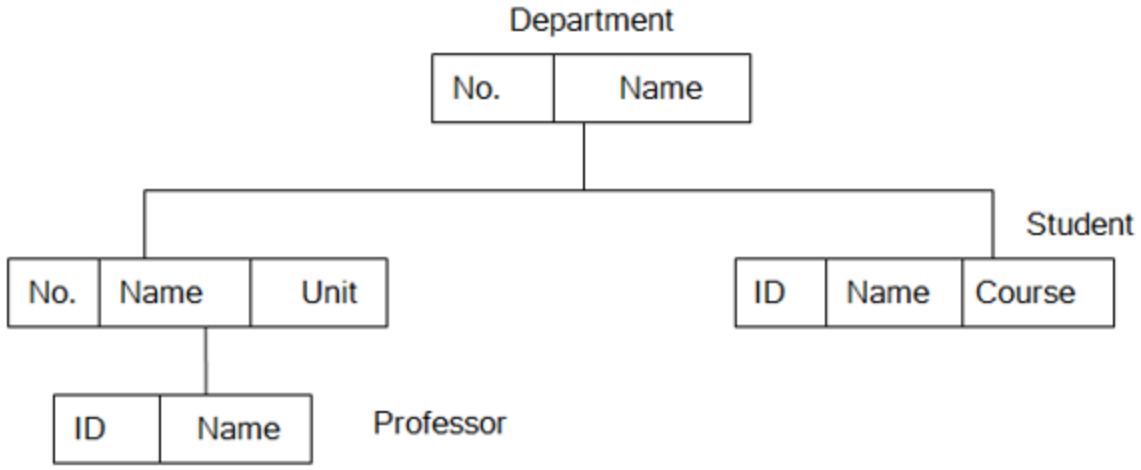
## डाटाबेस के प्रकार (Types of Database)

डाटाबेस मुख्य रूप से तीन प्रकार का होता है, जो कि निम्नलिखित हैं-

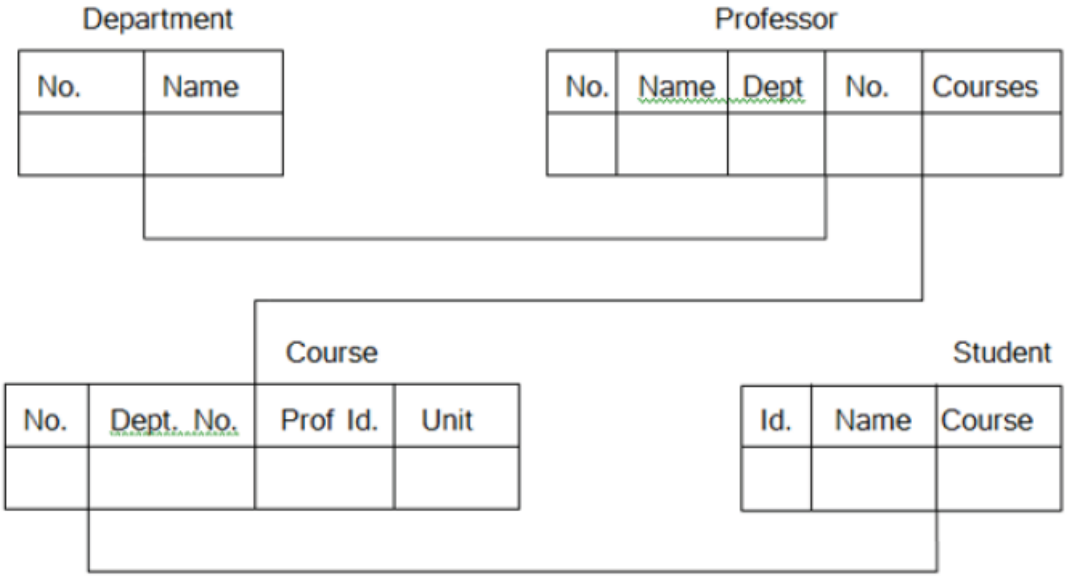
1. **Network Database -** इस प्रकार के डाटाबेस में डाटा को रिकॉर्ड के रूप में दर्शाया जाता है और डाटा के बीच संबंध लिंक के रूप में दर्शाया जाता है।




2. **Hierarchical Database -** इस तरह के डाटाबेस में डाटा को इसके साथ ट्री स्ट्रक्चर (पैरेंट चाइल्ड) के रूप में व्यवस्थित किया जाता है। नोड्स लिंक के माध्यम से जुड़े हुए हैं।




3. **Relational Database -** इस डाटाबेस को स्ट्रक्चरल डाटाबेस के रूप में भी जाना जाता है। जिसमें डाटा टेबल्स के रूप में संग्रहीत होता है। जहाँ स्तंभ (Column) और पंक्तियों (Rows) में संग्रहित किया जाता है।




## डाटाबेस सिस्टम संरचना (Database System Structure)

डाटाबेस सिस्टम को मॉड्यूल (Modules) में विभाजित किया गया है तथा हर मॉड्यूल पर System Control से सम्बन्धित जिम्मेदारी होती है। Database System को क्रियाओं के अनुरूप दो भागों में विभाजित किया गया है -

1. **स्टोरेज मैनेजर (Storage Manager)**
   स्टोरेज मैनेजर वह module है जो Low Level Data तथा Application Programs व Queries के मध्य पारस्परिक क्रिया (Interaction) करता है। स्टोरेज मैनेजर डाटा के Relational व अद्यतन (updation) के लिए उत्तरदायी होता है। स्टोरेज मैनेजर में निम्न अवयव होते हैं -
   a. **Authorization and integrity Manager -** इसमें डाटा की पूर्णता वाली शर्ते (integrity Constraints) तथा उपयोगकर्ता (users) की अधिकारिता (authority) को भी सुनिश्चित किया जाता है।
   b. **Transaction Manager -** यह Database की consistency के लिये उत्तरदायी होता है तथा Transaction को लगातार बिना व्यवधान निष्पादित (execute) करने का कार्य भी करता है।
   c. **File Manager -** इसके द्वारा मेमोरी में स्थान व डाटा स्ट्रक्चर को प्रदर्शित किया जाता है।
   d. **Buffer Manager -** डाटा को मुख्य मेमोरी में लाने के लिए उत्तरदायी होता है।
   स्टोरेज मैनेजर के द्वारा विभिन्न Data Structure को Implement किया जाता है-
      - **Data files** - जिसमें डाटा Store रहता है।
      - **Data Dictionary** - जिसमें डाटा के बारे में डाटा को रखा जाता है।
      - **Indices** - इससे डाटा को त्वरित प्राप्त किया जा सकता है।
2. **क्वेरी प्रोसेसर (Query Processor)**
   - **DDL Interpreter-** जो DDL निर्देशों को interpret करता है तथा उन निर्देशों की परिभाषा को Data Dictionary में संधारित करता है।
   - **DML Compiler-** जो DML निर्देशों को Low-level instruction में परिवर्तित करने का कार्य करता है।
   - **Query Evaluation Engine -** जो DML Compiler द्वारा परिवर्तित Low-level निर्देशों को execute करने का कार्य करता है।

**कम्प्यूटरीकृत डाटाबेस की आवश्यकता (Need of Computerized Database) -** हाथ से बनाए गए डाटाबेस सी समस्याएँ होती हैं जैसे कि-में बहुत (हस्तचालित डाटाबेस)-

- नया डाटा जोड़ने की समस्या।
- डाटा को बदलने की समस्या।
- डाटा को अपनी शर्तों के अनुसार प्राप्त करने की समस्या आदि।

इन सभी समस्याओं को दूर करने के लिए कम्प्यूटरीकृत डाटाबेस का निर्माण किया गया। इसमें सभी सूचनाएँ कम्प्यूटर पर रखी जाती है और कम्प्यूटर की सहायता से ही उनका रख-रखाव तथा प्रोसेसिंग की जाती है। कम्प्यूटर पर डाटाबेस बनाने की कई प्रोसेसिंग की जाती है। कम्प्यूटर पर डाटाबेस बनाने के कई कारण हैं, जो निम्नलिखित हैं –

1. कम्प्यूटर पर बड़े आकार का डाटाबेस सरलता से बनाया जा सकता है, क्योंकि उसमें डाटा को संग्रहित करने की क्षमता अधिक होती है।
2. कम्प्यूटर की कार्य करने की गति तेज होने के कारण कितने भी बड़े डाटाबेस में से कोई भी इच्छित सूचना निकालना और डाटाबेस पर विभिन्न क्रियाएँ करना आदि कार्य बहुत कम समय में ही सम्पन्न हो जाते हैं। इतना ही नहीं तेज गाति के कारण उस पर कोई लम्बीचौड़ी रिपोर्ट निकालना और छापना- मिनटों का कार्य होता है।
3. इसमें हस्तचालित (Manual) डाटाबेस की तुलना में बहुत कम खर्च लगता है।

**डाटाबेस के अवयव (Components of Database) -** एक डाटाबेस विभिन्न प्रकार के अवयवों से मिलकर बना होता है। डाटाबेस का प्रत्येक अवयव ऑब्जेक्ट (object) कहलाता है।

प्रत्येक डाटाबेस फाइल में आप अपने डाटा को विभिन्न सारणियों (Tables) में विभाजित कर सकते हैं। फॉर्म (Form) के माध्यम से सारणी के डाटा को देख सकते हैं। नया डाटा जोड़ सकते हैं तथा अपडेट (Update) भी कर सकते हैं। क्वेरेज (Queries) के माध्यम से आवश्यकतानुसार सारणियों में से डाटा को खोज सकते हैं तथा पुनः प्राप्त कर सकते हैं और रिपोर्ट (Report) के माध्यम से डाटा का विश्लेषण (Analyses) तथा डाटा को एक विशेष लेआउट (layout) में प्रिन्ट कर सकते हैं।

**डाटाबेस के अवयवों का विस्तारपूर्वक वर्णन निम्नलिखित हैं-**

1. **सारणी (Table) -** सारणी, स्तम्भ तथा पंक्तियों के कटाव से बने सैल (Cells) से मिलकर बनी होती है, यही सेल सारणियों में डाटा को स्टोर करने के लिए प्रयोग की जाती है।
   इन सारणियों पर विभिन्न प्रकार के ऑपरेशन, जैसे कि डाटा को स्टोर करना, निस्पन्दन (Filtering) करना, पुनः प्राप्त करना, डाटा का सम्पादन करना आदि किए जा सकते हैं। मुख्य रूप से, सारणी फील्ड तथा रिकॉर्ड से मिलकर बनी होती है जिनका विवरण निम्नलिखित हैं –
   (i) **फील्ड (Field) -** सारणी के प्रत्येक स्तम्भ को फील्ड कहते हैं। प्रत्येक फील्ड का एक निश्चित नाम होता है, जिसमे उसे पहचाना जाता है। प्रत्येक फील्ड का नाम उस फील्ड में स्टोर होने वाले डाटा के प्रकार को बताता है। उदाहरण के लिए विद्यार्थी का नाम, शहर, देश, टेलीफोन नम्बर आदि फील्ड के नाम हो सकते हैं।
   (ii) **रिकॉर्ड (Record) -** सारणी की प्रत्येक पंक्ति (row) को रिकॉर्ड कहा जा सकता है। दूसरे शब्दो में एक रिकॉर्ड एक एंटिटी (जैसे कि वस्तु, व्यक्ति शादि) से सम्बन्धित सभी फील्डों में उपस्थित डाटा का संग्रह होता है।
2. **क्वेरेज (Queries) -** किसी सारणी या डाटाबेस से आवश्यकतानुसार डाटा को निकालने के लिए जो आदेश दिया जाता है, use क्वेरी कहा जाता है। किसी क्वेरी के उत्तर में जो सूचनाएँ या रिकॉर्ड डाटाबेस से निकाले जाते हैं, उसे उस क्वेरी का डायनासेट (Dynaset) कहते हैं।
3. **फॉर्म (Forms) -** फॉर्म आपकी स्क्रीन पर ऐसी विण्डो होती है, जिसकी सहायता से आप किसी सारणी में भरे गए डाटा को देख सकते हैं, सुधार सकते हैं और नया डाटा जोड़ भी सकते हैं, सामान्यतः फॉर्म एक समय पर एक रिकॉर्ड को देखने तथा सुधारने के लिए प्रयोग किया जाता है।
4. **रिपोर्ट (Reports) -** सरल शब्दों में कोई रिपोर्ट एक ऐसा डायनासेट है, जिसे कागज पर छापा गया हो, आप किसी डायनासेट की सूचनाओं को किन्हीं आधारों पर समूहबद्ध कर सकते हैं।

## डाटाबेस मैनेजमेंट सिस्टम की विशेषताएँ (Characteristic of DBMS)

- इसमें डाटा की पुनरावर्ती (redundancy) को नियंत्रण (control) किया जा सकता है।
- डाटाबेस मैनेजमेंट सिस्टम में डाटा को साझा किया जा सकता है।
- डाटाबेस मैनेजमेंट सिस्टम में सुरक्षा (security) का पूरा ध्यान रखा जाता है।
- डाटाबेस मैनेजमेंट सिस्टम में प्रोसेसिंग की गति अच्छी है।
- डाटाबेस मैनेजमेंट सिस्टम में डाटा स्वतंत्र (independent) होता है।

## डाटाबेस के अनुप्रयोगी क्षेत्र (Application Areas of Database)

डाटाबेस का उपयोग विभिन्न क्षेत्रों में किया जाता है। जिनमें से कुछ निम्नलिखित है-

- **बैंकिग** के क्षेत्र में ग्राहकों की पर्सनल सूचना, उनके खातों की सूचना, लोन (Loans) आदि की सूचना रखने के लिए। विश्वविद्यालयों में विद्यार्थियों की सूचना, उनके अंक, कोर्स रजिस्ट्रेशन की सूचना आदि रखने के लिए।
- **एयरलाइन (Airline)** में रिजर्वेशन (Reservation) तथा कार्यक्रम की सूचना आदि के लिए।
- **क्रेडिट कार्ड** के लेन-देन में (Credit-card Transaction) क्रेडिट कार्ड के द्वारा खरीदारी तथा मासिक लेन-देन की रिपोर्ट तैयार करने के लिए।
- **संचार** के क्षेत्र में कॉल (Call) की मासिक रिकॉर्ड रखने के लिए, मासिक बिल बनाने के लिए।
- **विक्रय (Sale)** के क्षेत्र में व्याहकों, उत्पादों तथा खरीदारी की सूचना रखने के लिए।
- **वित्तीय (Finance)** क्षेत्र में बिक्री तथा खरीद के बारे में जानकारी संग्रहित करने के लिए।
- **एच. आर. (Human Resource)** के क्षेत्र में कर्मचारियों, उनके वेतन, टैक्स आदि के बारे में जानकारी संग्रहित करने के लिए।

## DBMS के लाभ (Advantages of DBMS)

DBMS के कई लाभ हैं- जो निक्नलिखित हैं –

- **डाटा के दोहराव में कमी (Reduction in Data Repetition) -** अच्छी तरह व्यवस्थित किए गए डाटाबेस में सामान्यतः डाटा का कोई दोहराव नहीं होता। समस्त डाटा को एक जगह रखे जाने के कारण हर सूचना को केवल एक बार स्टोर किया जाता है।
- **डाटा की स्थिरता (Data Consistency) -** डाटा के एक ही स्थान पर केन्द्रित (centralized) होने के कारण डाटा की स्थिरता बनी रहती है, क्योंकि उसमें एक ही सूचना के दो मानों की सम्भावना समाप्त हो जाती है। डाटा स्थिर तब होता है जब डाटा दो जगह रखा गया हो और केवल एक जगह सुधारा गया हो।
- **डाटा की साझेदारी (Data Sharing) -** डाटा की साझेदारी करके एक समय पर कई प्रोग्राम डाटा का प्रयोग कर सकते हैं। जिससे प्रोग्रामों को अपना डाटाबेस तैयार करने की आवश्यकता नहीं होती और बहुत-सा समय और परिश्रम बच जाता है।
- **डाटा की सुरक्षा (Security of Data) -** डाटाबेस प्रबन्धन प्रणाली (DBMS) डाटा को निषिद्ध उपयोगकर्ताओं (banned users) तथा अवैध परिवर्तन (invalid change) से बचाता है। यह केवल अधिकृत उपयोगकर्ताओं (authorized users) को डाटा का प्रयोग करने की अनुमति प्रदान करता है।
- **डाटा की सम्पूर्णता (Data Integrity) -** डाटा की सम्पूर्णता, डाटा की समय पूर्णता (Overall Completeness), सटीकता (Accuracy) तथा निरन्तरता (Consistency) को सन्दर्भित करती है। यह एक डाटा रिकॉर्ड के दो अपडेट्स (Updates) के बीच परिवर्तन के अभाव को दर्शाता है। यह दर्शाता है कि डाटाबेस में स्टोर डाटा बिल्कुल सही है और नवीनतम है।

## DBMS की सीमाएँ (Limitation of DBMS)

DBMS के कई लाभ हैं, लेकिन साथ ही इसकी कुछ सीमाएँ भी हैं जो निम्नलिखित हैं -

- **हार्डवेयर और सॉफ्टवेयर की लागत (Cost of Hardware and Software) -** सॉफ्टवेयर को चलाने के लिए डाटा को तीव्र गति से प्रोसेस करने वाले प्रोसेसर (Processor) और अधिक क्षमता वाली मेमोरी (Memory) की आवश्यकता होती है, जिनकी लागत अधिक होती है।
- **कठिनता (Complexity) -** एक डाटाबेस प्रबन्धन प्रणाली (DBMS) के अच्छे कार्य करने की क्षमता की पूर्व-कल्पना करना उस DBMS सॉफ्टवेयर को कठिन बना देती है। डाटाबेस प्रबन्धन प्रणाली को समझने की विफलता एक संगठन Organization के लिए गम्भीर परिणामों का कारण बन सकती है।
- **कर्मचारियों के प्रशिक्षण की लागत (Cost of Staff Training) -** अधिकतर DBMS सॉफ्टवेयर अत्यन्त जटिल (complex) होते हैं, इसलिए उपयोगकर्ताओं को डाटाबेस का प्रयोग करने के लिए एक प्रशिक्षण देने की आवश्यकता होती है। इस प्रकार, DBMS सॉफ्टवेयर चलाने के लिए संगठन को कर्मचारियों के प्रशिक्षण के लिए एक बड़ी राशि का भुगतान करना पड़ता है।

- **टेक्निकल स्टाफ की नियुक्ति (Appointing Technical Staff) -** एक संगठन में डाटाबेस के लिए प्रशिक्षित टेक्निकल पर्सन (Trained Technical Staff) जैसे कि डाटाबेस व्यवस्थापक (Database Administrator), एप्लीकेशन प्रोग्रामर (Application Programmers) आदि की आवश्यकता होती है, जिसके लिए संगठन को इन व्यक्तियों को एक अच्छे वेतन का भुगतान करना पड़ता है जिससे प्रणाली की लागत बढ़ जाती है।
- **डाटाबेस की विफलता (Database Failure) -** अधिकांश संगठन में सभी डाटा एक ही डाटाबेस में एकीकृत होता है। यदि पावर बन्द हो जाने के कारण डाटाबेस विफल हो जाता है या डाटाबेस स्टोरेज डिवाइस पर ही विफल (Fail) हो जाता है। तब हमारा सभी मूल्यवान (Valuable) डाटा लुप्त (Loss) हो सकता है या हमारी पूरी प्रणाली बन्द हो सकती है।

**रिलेशनल डाटाबेस (Relational Database) -** रिलेशनल डाटाबेस में डाटा को द्वि आयामी सारणियों (2-Dimensional Tables) के रूप में संग्रहित किया जाता है। इन सारणियों को रिलेशन (Relation) भी कहा जाता है। रिलेशन डाटाबेस के रख-रखाव के लिए रिलेशनल डाटाबेस प्रबन्धान प्रणाली (Relational Database Management System-RDBMS) की आवश्यकता होती है। RDBMS, DBMS का ही एक प्रकार है। रिलेशनल डाटाबेस की मुख्य विशेषता यह है कि एकल डाटाबेस में एक से अधिक सारणियों को संग्रहित किया जा सकता है और ये सारणियाँ आपस में सम्बन्धित होती हैं।

## सम्बन्धित पदावली (Related Terminology)

1. **रिलेशन (Relation) -** रिलेशन के अन्तर्गत एक टेबल (Table) तैयार की जाती है जो एक सिक्वेन्शियल फाइल को निरूपित करती है, जिसमें टेबल की पंक्तियाँ (Rows) फाइल के रिकॉर्ड को इंगित करती हैं एवं स्तम्भ (Column) रिकॉर्ड के फील्ड को दर्शाता है। ये टेबल्स रिलेशन ही होते हैं। रिलेशन को उच्च स्तरीय फाइल्स के रूप में समझा जाता है, –
   - प्रत्येक रिलेशन में एक ही तरह के रिकॉर्ड होते हैं।
   - किसी दिए गए रिलेशन में प्रत्येक रिकॉर्ड के फील्डों की संख्या समान होती हैं।
   - प्रत्येक रिकॉर्ड का एक अलग पहचानने वाला (identifier) होता है।
   - रिलेशन के अन्दर रिकॉर्ड किसी विशेष क्रम में व्यवस्थित होते हैं।

इसके लिए निम्नलिखित उदाहरण पर विचार कीजिए -

**Relation - Part**

| P# | P Name | Colour | Weight | City |
|---|---|---|---|---|
| P1 | Nut | Red | 12 | London |
| P2 | Bolt | Green | 15 | Paris |
| P3 | Screw | Blue | 18 | Rome |
| P4 | Screw | Red | 14 | London |
| P5 | Cam | Blue | 19 | Paris |

2. **ट्यूपल (Tuple) -** रिलेशन में प्रत्येक रिकॉर्ड को ट्यूपल कहा जाता है। उदाहरण के लिए, दिए गए रिलेशन Parts में पाँच ट्यूपल हैं। उनमें से एक ट्यूपल (P2, Bolt, Green, 15, Paris) है जो एक Part के विषय में एक विशेष सूचना है।
3. **एट्रिब्यूट (Attribute) -** रिलेशन के सन्दर्भ में प्रत्येक कॉलम (फील्ड) को एट्रिब्यूट कहते हैं। उदाहरण के लिए, दिए गए रिलेशन Parts में पाँच एट्रिब्यूट्स (P4, P Name, Colour, Weight, City) हैं। जिनमें से प्रत्येक कॉलम एक Part के विषय में सूचना प्रदान करता है।
4. **डोमेन (Domain) -** रिलेशन के सन्दर्भ में डोमेन मानों का एक समूह होता है जिससे किसी कॉलम में दिए गए वास्तविक मानों को व्युत्पन्न किया जा सकता है।
   उदाहरण के लिए, हम निम्न रिलेशन पर विचार कर सकते हैं।

**Relation – S**

| S# | S Name | Status | City |
|---|---|---|---|
| S1 | Amar | 30 | Paris |
| S2 | Mohan | 20 | New Delhi |
| S3 | Ram | 10 | London |

**Relation – P**

| P# | P Name | Status | Quantity |
|---|---|---|---|
| P1 | Nut | 12 | A |
| P2 | Bolt | 15 | B |
| P3 | Screw | 25 | C |

**C Relation – SP**

| P# | S# | Quantity |
|---|---|---|
| P1 | S1 | 300 |
| P2 | S2 | 400 |
| P3 | S3 | 200 |
| P2 | S2 | 300 |
| P1 | S1 | 200 |

यहाँ SP टेबल के P# कॉलम में जो मान दिए गए हैं उन्हें P टेबल से व्युत्पन्न किया गया है एवं SP टेबल के S# कॉलम में जो मान दिए गए हैं उन्हें S टेबल से व्युत्पन्न किया गया है। अतः यहाँ टेबल P एवं टेबल S एक डोमेन के रूप में हैं, जिनसे P# एवं S# मानों को व्युत्पन्न कर एक SP टेबल तैयार किया गया है।

5. **कार्डिनैलिटी (Cardinality) -** रिलेशन के सन्दर्भ में ट्यूपल रिकॉ)र्ड्स की कुल संख्या को कार्डिनैलिटी कहते हैं। अतः ( ऊपर वर्णित उदाहरण के लिए रिलेशन P की कार्डिनैलिटी 3, S की 3 एवं SP की 5 है।
6. **डिग्री (Degree) -** रिलेशन के सन्दर्भ में एट्रिब्यूट की कुल संख्या को रिलेशन की (फील्ड या कॉलम)डिग्री कहते हैं। अतः ऊपर दिए गए उदाहरण में रिलेशन P की डिग्री 4, S की 4 एवं SP की 3 है।

## रिलेशन डेटाबेस की (Relational Data base keys)

एक Relational Database में, सभी डेटा को टेबल में रखा जाता है, जो पंक्तियों और स्तंभों से बने होते हैं । प्रत्येक टेबल में एक या अधिक Column होते हैं और प्रत्येक Column को एक विशिष्ट डेटा प्रकार दिया जाता है, जैसे- integer number, a sequence of characters (for text), or a date आदि । टेबल में प्रत्येक पंक्ति में प्रत्येक कॉलम के लिए एक मान होता है।

### Relational Database key के प्रकार

1. **प्राइमरी की (Primary Key)**

   Primary Key, जिसे प्राईमरी कीवर्ड भी कहा जाता है। यह रिलेशनल डेटाबेस टेबल में एक Column है जो प्रत्येक रिकॉर्ड के लिए अद्वितीय (Unique) है । यह एक unique identifier है, जैसे ड्राइवर का लाइसेंस नंबर, क्षेत्र कोड वाला टेलीफोन नंबर या वाहन पहचान संख्या (वीआईएन) । एक relational database में केवल एक प्राईमरी - की होनी चाहिए। डेटा की प्रत्येक पंक्ति में एक प्राईमरी की मान होना चाहिए और कोई भी प्राईमरी - की में पंक्ति का मान Null नहीं हो सकता। एक Primary Key एक टेबल में एक फील्ड है जो विशिष्ट रूप से डेटाबेस टेबल में प्रत्येक पंक्ति / रिकॉर्ड की पहचान करती है।

   प्राथमिक कुंजी (Primary Key) के मुख्य लाभ निम्नलिखित हैं -

   (i) अद्वितीय डेटा की पहचान करने में मदद करता है, जैसे Customer ID आदि ।

   (ii) टेबल में रिकॉर्ड्स के दोहराव को रोकता है।

   (iii) केवल विशिष्ट रिकॉर्ड को अपडेट करने या हटाने में मदद करता है।

   (iv) एक Primary Key कॉलम में NULL मान नहीं हो सकते।

   (v) एक टेबल में केवल एक Primary Key हो सकती है, जिसमें एकल या एकाधिक फील्ड शामिल हो सकते हैं। जब Primary Key के रूप में कई Fields का उपयोग किया जाता है, तो उन्हें Composite Key कहा जाता है।

**Create Primary Key**

CUSTOMER Table में ID Attribute को प्राइमरी की के रूप में परिभाषित करने के लिए सिंटैक्स निम्न प्रकार है ।

```
CREATE TABLE CUSTOMERS (
ID          INT          NOT NULL,
NAME        VARCHAR (20) NOT NULL,
AGE         INT          NOT NULL,
ADDRESS CHAR (25),
SALARY DECIMAL (18, 2),
PRIMARY KEY (ID)
);
```

**Delete Primary Key**

**CUSTOMER Table में ID Attribute** को प्राइमरी - की के रूप में हटाने के लिए सिंटैक्स निम्न प्रकार से है

```
ALTER TABLE CUSTOMERS DROP PRIMARY KEY;
```

**2. कन्डीडेट की (Candidate Key)**

एक या एक से अधिक ऐसे एट्रिब्यूट जो डाटाबेस में यूनिक परिभाषित होते हैं एवं प्राइमरी की का चुनाव उन्हीं में से किया जाता है, वे सभी एट्रिब्यूट कन्डीडेट की कहलाते हैं। अर्थात् एक रिलेशनल डाटाबेस में, एक Candidate Key एक Single Column या एक से अधिक Columns का संयोजन हो सकता है, जिसे प्राइमरी - की के रूप में उपयोग किया जा सकता है। एक **"minimal super key"** को Candidate Key कहा जाता है।

विशेषताएँ

- प्राइमरी - की के विपरीत Candidate Key का एक NULL मान हो सकता है।
- Candidate Key प्राथमिक कुंजी हो भी सकती है और नहीं भी ।

**3. अल्टरनेट की (Alternate Key)**

Alternate Key या Secondary Keys वह Key है जिसे प्राइमरी - की के रूप में नहीं चुना जा सकता है लेकिन वह Candidate Key हैं। हालाँकि, इसे प्राइमरी - की के लिए Candidate Key माना जाता है।

प्राइमरी - की के रूप में नहीं चुनी गई Candidate Key को Alternate Key या Secondary Keys कहा जाता है ।

**उदाहरण –**

| S_ID | S_Enroll | S Name | S Email |
|---|---|---|---|
| 071 | 1115 | Garima | garima@gmail.com |
| 052 | 2215 | Monika | monika@gmail.com |
| 098 | 4028 | Muskan | muskan@gmail.com |

उपरोक्त उदाहरण में, S_ID, S_ Enroll और S_ Email से सभी Candidate Key हैं। उन्हें Candidate Key माना जाता है क्योंकि वे विशिष्ट रूप से Student Record की पहचान कर सकते हैं। प्राइमरी के रूप में Candidate Key में से किसी एक का चयन करें तथा बाकी दो Key Alternate या Secondary Key होंगी ।

**4. फोरेन की (Foreign Key)**

यह दो टेबल को आपस में लिंक करता है। एक Foreign Key एक टेबल में एक फील्ड (या फील्ड का संग्रह ) है, जो किसी अन्य टेबल में प्राईमरी - की को संदर्भित करती है

Foreign Key वाली टेबल को child table कहा जाता है, और प्राइमरी - की वाली टेबल को referenced या parent table कहा जाता है ।

**फोरेन - की की विशेषताएँ निम्नलिखित हैं-**

- एक टेबल में एक से अधिक फोरेन - की हो सकती हैं।

- Foreign Key Column Null मान स्वीकार करता है ।
- यह एक टेबल में Parent - Child Relationship का संबंध बना सकता है ।
- डुप्लिकेट मान Foreign Key कॉलम में संग्रहित किए जा सकते हैं।

**उदाहरण -**

**Persons Table**

| Person ID | Last Name | First Name | Age |
|---|---|---|---|
| 1 | Hansen | Ola | 30 |
| 2 | Svendson | Tove | 23 |
| 3 | Pettersen | Kari | 20 |

**Orders Table**

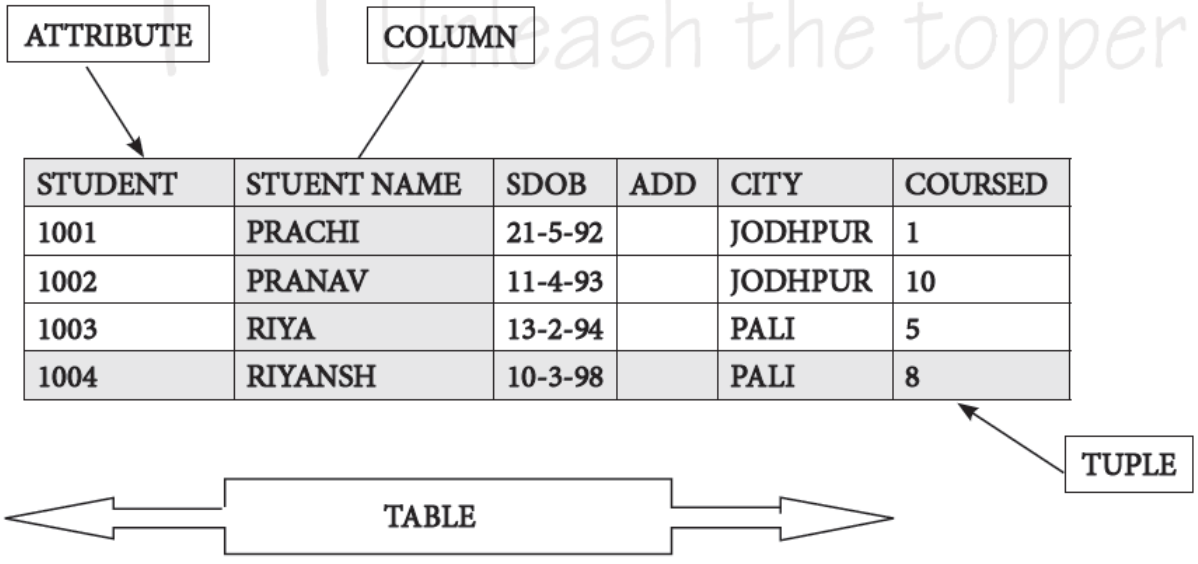
| OrderID | OrderNumber | PersonID |
|---|---|---|
| 1 | 77895 | 3 |
| 2 | 44678 | 3 |
| 3 | 22456. | 2 |
| 4 | 24562 | 1 |

ध्यान दें कि "Order" टेबल में "PersonID" कॉलम
"Person" टेबल में "PersonID" कॉलम को इंगित करता है ।
"PersonID" कॉलम "Person" टेबल में प्राइमरी - की है और "Order"
टेबल में "PersonID" कॉलम एक Foreign Key है ।

**ऐंटिटी रिलेशनशिप मॉडल (Entity Relationship Model) -** ऐंटिटी रिलेशनशिप मॉडल वास्तविक ऐंटिटी (real-world entity) और उनके बीच संबंधों की धारणा पर आधारित है। वास्तविक परिदृश्य को डाटाबेस मॉडल तैयार करते समय, ईआर मॉडल ऐंटिटी सेट, रिलेशन सेट, सामान्य एट्रिब्यूट और कंस्टेंट्स बनाता है।
ईआर मॉडल संकल्पनात्मक डिजाइन के लिए उपयोग में लिया जाता है। ER मॉडल निम्न पर आधारित है -

1. entities और attributes
2. entities के मध्य सम्बन्ध (Relationships)

**Entity -** ईआर मॉडल में ऐंटिटी में वास्तविक दुनिया के गुणों वाली एट्रिब्यूट होती है। हर एट्रिब्यूट में इसके मानों के समुच्चय (set) को डोमेन द्वारा परिभाषित किया जाता है। उदाहरण के लिए एक स्कूल डाटाबेस में छात्र एक ऐंटिटी के रूप में होता है। छात्र के भिन्न-भिन्न एट्रिब्यूट जैसे - नाम, उम्र , वर्ग, आदि होते हैं।

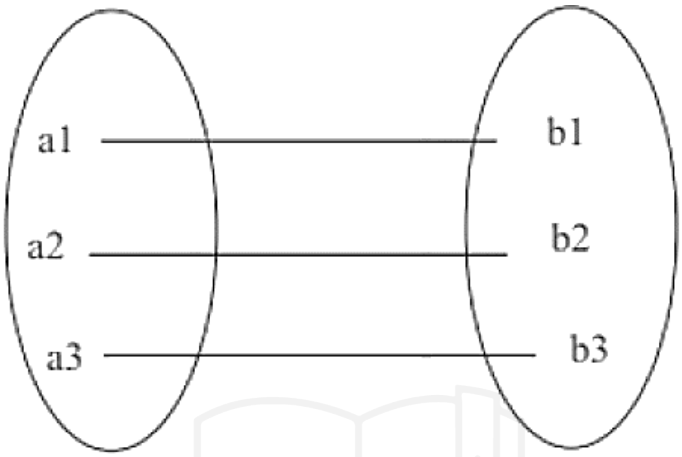


## ऐंटिटी रिलेशनशिप मॉडल

**रिलेशनशिप (Relationship) -** एक से अधिक एंटिटी के मध्य तार्किक (Logical) सम्बन्ध रिलेशनशिप कहलाता है। एंटिटी में संबंध विभिन्न तरीकों के साथ मैप की जाती है। मैपिंग कार्डीनलिटीज (mapping cardinalities) दो एंटिटी के मध्य सम्बन्धों की संख्या बताता है।

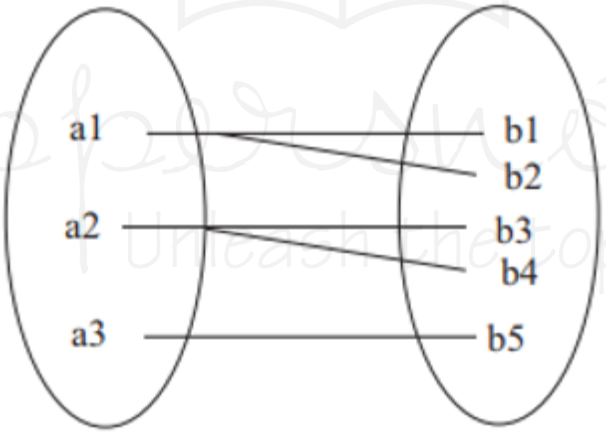
### मैपिंग कार्डीनलिटीज (mapping cardinalities) निम्न प्रकार की होती हैं –

1. **One to One**- Entity A का तत्व )element) अधिक से अधिक के B Entity के एक तत्व से Connected हो तथा B की एक Entity का तत्व अधिक से अधिक A Entity के एक तत्व से सम्बन्धित हो।



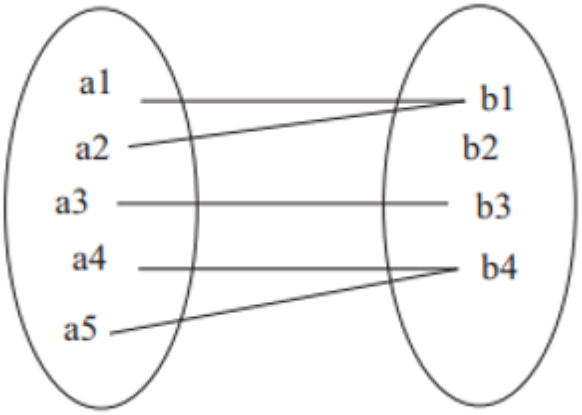

**वन टू वन रिलेशनशिप में**

2. **One to Many**- Entity के तत्व B Entity के एक से अधिक तत्व से सम्बन्धित हो सकते हैं। परन्तु B Entity के तत्व अधिक से अधिक A Entity के एक ही तत्व से सम्बन्धित हो सकते हैं।



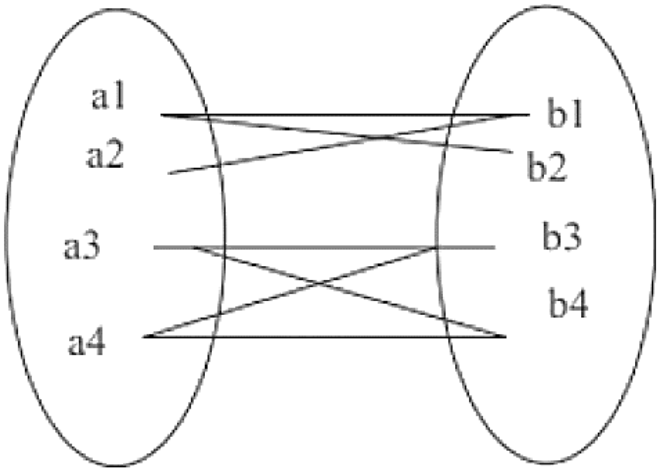

**वन टू मैनी रिलेशनशिप**

3. **Many to One**- A Entity के तत्व B Entity के एक ही तत्व से सम्बन्धित हो सकते हैं। परन्तु B Entity का एक तत्व एक A Entity के एक से अधिक तत्व से सम्बन्धित हो सकते हैं।




**मेनी टू वन रिलेशनशिप**

4. **Many to Many**- A Entity के तत्व B Entity के एक से अधिक तत्व से सम्बन्धित हो सकते हैं और B Entity के तत्व भी A Entity के एक से अधिक तत्व से सम्बन्धित हो सकते हैं।




**मेनी टू मेनी रिलेशनशिप**

## MySQL Introduction

MySQL एक डाटाबेस प्रबंधन प्रणाली है कि यह relational डाटाबेस को प्रबंधित करने के काम आता है । यह ओपन सोर्स सॉफ्टवेयर है ।

## MySQL Commands

## To login (from unix Shell)-

#[mysql dir]/bin/mysql- h hostname - u username - p password

यह Command उन Systems के लिए है जो unix operating system पर काम करते हैं।

## To login from windows-

mysql dir/bin/mysql.exe- h hostname - u username - p password

यह Command उन Systems के लिए हैं जो windows, operating system पर काम करते हैं।

## To create a database

mysql > create database (database name);

यह Command एक नया database बनाने के लिए प्रयोग की जाती है । यहाँ दो keywords का प्रयोग किया गया हैं- "create" और "database" और (dataname name) की जगह आपको अपने नए database (जो आप बनाना चाह रहे हैं) का नाम देना होता है । For e.g. यदि हमें 'basic' नाम का एक database बनाना है तो हम लिखेंगे ।

mysql > create database basic

## To list all database on the server-

Mysql > show databases;

यह Command server पर मौजुद सारे database की list देखने के लिए प्रयोग की जाती हैं ।

## To Switch to a database

mysql > use [db name]

यह Command किसी एक database से दूसरे database पर जाने के लिए प्रयोग की जाती है या हम यह भी कह सकते हैं कि किसी database को प्रयोग करने के लिए यह command प्रयोग की जाती है । यहाँ दो keywords का use किया गया हैं- "use" और "database" जबकि (db name) की जगह आप उस database का नाम लिखते हैं जो use करना चाह रहे हैं।

For. eg. हमारे पास दो database हैं - "basic" और "advanced" । वर्तमान में हम "basic" पर काम कर रहे है । यदि हमें "advanced" पर काम करना है तो हम command देंगे-

mysql > use advance

## To see all the tables in the database-

mysql > show tables;

यह Command किसी database में मौजूद सारी tables की list देखने के लिए की जाती है । इस Command के द्वारा आपको उस database में मौजूद सारी tables दिखाई देंगी जिस database पर आप वर्तमान में है ।

### To Create a table-

mysql > create table table name (column name data type (size), column name data type (size), ------)

For e.g.:

mysql > create table student (name varchar (50), Age int (10), roll-no. int (12), address varchar (100);

यह Command किसी database के अंदर एक नई tables create करने के लिए प्रयोग की जाती है । ऊपर जो उदाहरण दिया है उसमें हमने student नाम की एक table बनायी है जिसमें चार fields हैं-

name- जिसका size 50 है और डाटा type varchar है ।

age- जिसका size 10 है और datatype int है ।

roll-no- जिसका size 12 है और datatype int है ।

address- जिसका size 100 है और datype varchar है ।

## To see table's field formats-

mysql > describe [table name];

यह command किसी एक particular table की details देखने के लिए प्रयोग की जाती है । इसमें एक keyword का use किया गया है जिसका नाम है - "describe" और [table name] की जगह आप उस table का नाम लिखेंगे जिस table की detail आप देखना चाह रहे हैं।

जैसे - हम किसी table की detail देखना चाह रहे हैं जिसका नाम 'basic' है तो हम command लिखेंगे -

describe basic;

## To delete a database:-

mysql > drop database [database name]-

इस command का use किसी एक particular database को delete करने के लिए किया जाता है। इसमें दो keywords - "drop" और "database" का प्रयोग किया गया है और [database name] की जगह आप उस database का नाम लिखेंगे जिसे आप delete करना चाह रहे हैं।

जैसे - हमारे पास "company" नाम का एक database है जिसे हम delete करना चाह रहे हैं तो हम command देंगे -

drop database company;

### To delete a table-

mysql > drop table [table name];

यह कमांड किसी table को delete करने के लिए प्रयोग की जाती है। इसमें दो keywords-"drop" और "table" हैं और (table name) की जगह उस table का नाम आता है जिसे delete करना होता है।

जैसे अगर हमें 'basic' नाम की कोई table delete करनी हो तो हम लिखेंगे -

drop table basic;

## To empty a table-

truncate table [table name];

यह command किसी table को खाली करने के लिए प्रयोग की जाती है । यानी इस command के द्वारा table के अंदर का सारा data delete हो जाता है ।

## To show all data from a table-

mysql > select * from [table name];

किसी एक particular table के अंदर का सारा data show करने के लिए इस command का प्रयोग करते हैं। जैसे - अगर हमें 'basic' नाम की table का सारा data देखना हो तो हम लिखेंगे-

select * from basic;

## To return columns and column information-

mysql > show columns from [table name];

किसी table में कितने column हैं तथा उन columns की detail देखने के लिए इस command का प्रयोग करते हैं।

### To show particular rows with the given value-

mysql > select * from [table name] where [field name] = value";

इस command का प्रयोग किसी table की particular rows देखने के लिए किया जाता है । यहाँ हम "where" keyword के द्वारा condition देते हैं।

जैसे - हमारे पास एक 'users' नाम की table है और जिसमें एक column है "age" । अगर हमें उन सारे users की list देखनी हो या यूँ कहें कि हमें वे सारी "rows" देखनी हो जिनमें age की value 24 हो तो हम command देंगे-

Select * from basic where age = 24;

## To return number of rows-

mysql > select count (*) from [table name];

इस command का प्रयोग करके हम यह देख सकते हैं कि किसी table में वर्तमान में कितनी rows हैं।

## To delet a row from a table-

mysql > delete from [table name] where [field name] = 'field value';

इस command का प्रयोग किसी table के अंदर की किसी particular row या rows को delete करने के लिए किया जाता है । जैसे- किसी 'basic' नाम की table में एक column है age। यदि ऐसी सारी rows को delete करना हो जहाँ age की value 25 हो तो command लिखेंगे -

Delete from basic where age = 25;

## To update database permissions / privileges:-

mysql > flush privileges;

इस command का प्रयोग database की permissions को change या update करने के लिए किया जाता है । जैसे- यदि हम update command के द्वारा कोई updation कर रहे हैं लेकिन जो changes किए हैं वो load नहीं हो रहे तो हम इस command का प्रयोग करेंगे ।

## To delete a column-

mysql > alter table [table name] drop column [column name];

यह Command किसी table से किसी column को delete करती है । जैसे - यदि हमें 'basic' नाम की table में 'address' नाम का column delete करना हो तो command होगी-

alter table basic drop column address;

## To add a new column-

mysql > alter table [table name] add column [new column name] varchar (20);

यह command किसी table में एक नया column (या field) जोड़ने के लिए प्रयोग की जाती है । जैसे- हमें अगर 'basic' नाम की table में 'address' नाम की एक field जोड़ी हो (जिसका datatype varchar और size 50 है) तो command लिखेंगे-

alter table basic add column address varchar (50);

### To change a column's name-

mysql > alter table [table name] change (old column name) (new column name) varchar(50);

यह command किसी table के किसी column का नाम change करने के लिए प्रयोग की जाती है । जैसे - हमारे पास कोई basic नाम की table है जिसमें address नाम की एक fleld है । अगर हमें address नाम की field का नाम change करके useraddress करना हो तो हम command लिखेंगे-

alter table basic change address newaddress;

## MySQL Functions

MySQL Functions के मुख्यतया दो प्रकार होते हैं-

1. MySQL Numeric Functions
2. MySQL String Functions