ऑन लाइन पाठ्य सामग्री

1PGDCA3(B)
DATABASE USING MS-ACCESS
(Elective-I)

इकाई - एक

मनोज निवारिया
सहा. प्राध्यापक, कम्प्यूटर विज्ञान एवं अनुप्रयोग
माखनलाल चतुर्वेदी राष्ट्रीय पत्रकारिता एवं संचार विश्वविद्यालय, भोपाल

माखनलाल चतुर्वेदी राष्ट्रीय पत्रकारिता एवं संचार विश्वविद्यालय
बी-38, विकास भवन, एम.पी. नगर, झोन - I, भोपाल

डेटाबेस यूजिंग एमएस एक्सेस

1. डाटा मॉडल

आधारभूत डेटाबेस की संरचना को समझना डेटा मॉडल है । डेटा, डेटा रिलेशनशिप, डेटा शब्दार्थ (data semantics), और स्थिरता बाधाओं (consistency constraints) का वर्णन करने के लिए वैचारिक उपकरणों का एक संग्रह है ।

एक डेटा मॉडल की अवधारणा को समझाने के लिए, हम दो डेटा मॉडल को विस्तार से समझेंगे : इकाई-संबंध (Entity-Relationship) मॉडल और संबंधपरक मॉडल (Relational Model) । दोनों तार्किक स्तर पर एक डेटाबेस के डिजाइन का वर्णन करने का तरीका प्रदान करते हैं।

1.1 इकाई-संबंध मॉडल / The Entity-Relationship Model

इकाई-संबंध (ई-आर) डेटा मॉडल एक वास्तविक दुनिया की धारणा पर आधारित है जिसमें मूल वस्तुओं का एक संग्रह होता है, जिसे इकाई (एंटिटी) कहा जाता है, और इन वस्तुओं के बीच संबंधों का एक संग्रह होता है । एक एंटिटी वास्तविक दुनिया में एक "बात" या "वस्तु" है जो अन्य वस्तुओं से अलग है। उदाहरण के लिए, प्रत्येक व्यक्ति एक एंटिटी है, और बैंक खातों को एंटिटी के रूप में माना जा सकता है।

किसी डेटाबेस में एंटिटीज़ को विशेषताओं (**attributes**) के एक सेट द्वारा वर्णित किया गया है। उदाहरण के लिए, एट्रिब्यूट खाता-संख्या और शेषराशि एक बैंक में किसी विशेष खाते का वर्णन कर सकते हैं, और खाता एंटिटीसेट के एट्रिब्यूट हो सकते हैं । इसी तरह, एट्रिब्यूट ग्राहक-नाम, ग्राहक-सड़क का पता और ग्राहक-शहर एक ग्राहक एंटिटी का वर्णन कर सकते हैं।

ग्राहकों को विशिष्ट रूप से पहचानने के लिए एक अतिरिक्त एट्रिब्यूट ग्राहक-आईडी का उपयोग किया जाता है (क्योंकि एक ही नाम, सड़क का पता और शहर वाले दो ग्राहक होना संभव हो सकता है)।

एक अद्वितीय ग्राहक पहचानकर्ता, प्रत्येक ग्राहक को सौंपा जाना चाहिए। जैसे भारत में, कई उद्यम/संस्थाएं ग्राहक की पहचानकर्ता के रूप में एक व्यक्ति की आधार संख्या (uidai, भारत सरकार प्रत्येक व्यक्ति को प्रदान करता है) का उपयोग करते हैं।

एक संबंध (Relationship) कई एंटिटीयों के बीच एक संबंध/जुडाव है। उदाहरण के लिए, एक जमाकर्ता रिलेशनशिप प्रत्येक खाते (जो उसके पास है) के साथ एक ग्राहक को जोड़ता है ।

एक ही प्रकार की सभी एंटिटीयों (संस्थाओं) के सेट और एक ही प्रकार के सभी रिलेशनशिप (संबंधों) के सेट को क्रमशः एक एंटिटी सेट (इकाई सेट) और रिलेशनशिप सेट (संबंध सेट) कहा जाता है।

एक डेटाबेस के समग्र तार्किक संरचना (Logical Structure) (स्कीमा / Schema) को ई-आर आरेख (E-R Diagram) द्वारा रेखांकन द्वारा (Graphically) व्यक्त किया जा सकता है।

1.2 संबंधपरक मॉडल (Relational Model)

संबंधपरक मॉडल डेटा और उन डेटा के बीच संबंधों (Relationships) को दर्शाने के लिए तालिकाओं (Tables) के संग्रह का उपयोग करता है। प्रत्येक तालिका (Table) में कई स्तंभ (Columns) होते हैं और प्रत्येक स्तंभ (Column) का एक अद्वितीय नाम होता है।

डेटा को एक ऐसे रिलेशन में व्यवस्थित किया जाता है जिसे दो आयामी तालिका (two dimensional table) में दर्शाया जाता है। डेटा को टपल्स (tuples) के रूप में टेबल में डाला जाता है (टपल्स मतलब पंक्तियां / rows) । एक टपल का गठन एक या एक से अधिक ऐट्रिब्यूट्स द्वारा किया जाता है । तालिका में किसी भी संख्या में टपल्स हो सकते हैं, लेकिन सभी टपल में अलग-अलग मानों (values) के साथ निश्चित और समान विशेषताएं (ऐट्रिब्यूट्स) होती हैं। संबंधपरक मॉडल (Relational Model) को डेटाबेस में इस प्रकार लागू किया जाता है, जहां एक संबंध (Relation) एक तालिका (Table) द्वारा दर्शाया जाता है । एक टपल (tuple) को एक पंक्ति (Row) द्वारा दर्शाया जाता है । टेबल के एक कॉलम द्वारा एक एट्रिब्यूट को दर्शाया जाता है । एट्रिब्यूट का नाम ही कॉलम का नाम होता है जैसे नाम, शहर आदि । एट्रिब्यूट की वैल्यू ही किसी रो में कॉलम के लिए वैल्यू होती है। नियमों और शर्तों (Constraints) को टेबल पर लागू किया जाता है और तार्किक स्कीमा (Logical Schema) बनाते हैं। टेबल से किसी विशेष पंक्ति / टपल के चयन को सुविधाजनक बनाने के लिए कॉलम के नामों (मतलब ऐट्रिब्यूट्स) का उपयोग किया जाता है । और पंक्तियों (rows) के चयन के लिए कुछ फ़ील्ड्स (fields) को विशिष्ट रूप से अनुक्रमित (indexes) के रूप में उपयोग करने के लिए परिभाषित किया जाता है। इससे आवश्यक डेटा खोजने में मदद मिलती है । सभी संबंधपरक बीजगणित संचालन (Relational Algebra Operations), जैसे कि चयन (Select), अंतर्ग्रहण (Intersection), उत्पाद(Product), संघ (Union), अंतर (Difference), परियोजना (Project), सम्मिलित (Join), विभाजन (Division), विलय (Merge) आदि भी संबंधित डेटाबेस मॉडल (Relational Database Model) पर किए जा सकते हैं। रिलेशनल डेटाबेस मॉडल पर

संचालन (Operations) विभिन्न Conditional expressions, विभिन्न Key Attributes, पूर्व-परिभाषित Constraints आदि की सहायता से किया जाता है।

2. इकाई-संबंध मॉडल (Entity Relationship Model) :

एंटिटी-रिलेशनशिप (ईआर) मॉडल को मूल रूप से 1976 में पीटर द्वारा नेटवर्क और रिलेशनल डेटाबेस विचारों को एकजुट करने के लिए प्रस्तावित किया गया था। सीधे तौर पर कहा गया है कि ई-आर मॉडल एक वैचारिक डेटा मॉडल है, जो वास्तविक दुनिया को संस्थाओं (एंटिटी) और रिश्तों (रिलेशन) के रूप में देखता है। इस मॉडल का एक मूल घटक इकाई-संबंध चित्र (एंटिटी-रिलेशनशिप डायग्राम) है, जो डेटा ऑब्जेक्ट्स को चित्र रूप से प्रस्तुत करने के लिए उपयोग किया जाता है। चूंकि चेन ने उनके पेपर में लिखा था कि यह मॉडल बढ़ाया गया है और आज इसका इस्तेमाल आमतौर पर डेटाबेस डिज़ाइन के लिए किया जाता है । एक डेटाबेस डिज़ाइनर के लिए, ER मॉडल की उपयोगिता है :

यह संबंधपरक मॉडल (Relational Model) को अच्छी तरह से मैप करता है। ई-आर मॉडल में उपयोग किए गए निर्माण (Constructs) आसानी से रिलेशनल टेबल में बदल सकते हैं।

न्यूनतम प्रशिक्षण के साथ इसे समझना सरल और आसान है। इसलिए, इस मॉडल का उपयोग डेटाबेस डिजाइनर द्वारा डिजाइन को अंतिम उपयोगकर्ता (End User) तक पहुंचाने के लिए किया जा सकता है ।

इसके अलावा, एक विशिष्ट डेटाबेस प्रबंधन सॉफ्टवेयर (Specific Database Management Software) में डेटा मॉडल को लागू करने के लिए डेटाबेस डेवलपर द्वारा डिजाइन योजना (Design Plan) के रूप में इस मॉडल का उपयोग किया जा सकता है ।

2.1 ई-आर मॉडलिंग के बुनियादी निर्माण / Basic Constructs of E-R Modeling :

ई-आर मॉडल, एंटिटीयों के निर्माण और एंटिटीयों के बीच सम्बन्ध के रूप में वास्तविक दुनिया को देखता है।

2.1.1 संस्थाओं / Entities (एंटिटीस) :

एंटिटीज प्रमुख डेटा ऑब्जेक्ट हैं जिनके बारे में जानकारी एकत्र की जानी है। एंटिटीज आमतौर पर पहचानने योग्य अवधारणाएँ (recognizable concepts) होती हैं, या तो ठोस (concrete) या अमूर्त (abstract), जैसे व्यक्ति, स्थान, चीज़ें, या घटनाएँ जिनकी डेटाबेस में

प्रासंगिकता होती है। एंटिटीज के कुछ उदाहरण कर्मचारी, परियोजनाएं, बिल हैं। एक एंटिटी, संबंधपरक मॉडल (Relational Model) में एक तालिका (Table) के अनुरूप है।

एंटिटीस को स्वतंत्र या निर्भर के रूप में वर्गीकृत किया जाता है (इनके लिए क्रमशः मजबूत (Strong) और कमजोर (Weak) शब्दों का प्रयोग होता हैं)। एक स्वतंत्र एंटिटी वह है जो पहचान के लिए दूसरे पर निर्भर नहीं होती है । एक निर्भर / आश्रित एंटिटी वह है जो पहचान के लिए दूसरे पर निर्भर होती है।

एक इकाई उपस्थिति (Entity occurence) (जिसे एक उदाहरण (instance) भी कहा जाता है) एक एंटिटी की एक व्यक्तिगत उपस्थिति है। एक उपस्थिति संबंधपरक टेबल (Relational Table) में एक पंक्ति (Row) के अनुरूप है।

विशेष एंटिटी के प्रकार (Special Entity Types) :

एसोसिएटिव एंटिटीस (Assocoative Entities), दो या दो से अधिक एंटिटीस को जोड़ने के लिए उपयोग की जाने वाली एंटिटीस हैं, जिससे कई-से-कई संबंधों (Many-to-Many Relationship) बनाये जा सकें । इनको इंटरसेकशन एंटिटीस भी कहा जाता है ।

उपप्रकार इकाइयाँ (Subtypes Entities) का उपयोग सामान्यीकरण पदानुक्रम (Generalization Hierarchies) में अपने पैरेंट एंटिटी (Parent Entity) के उदाहरणों (instances) के सबसेट (subset) का दर्शाने के लिए किया जाता है । पैरेंट एंटिटी (Parent Entity) को सुपरपाइप (Supertype) कहा जाता है । लेकिन इस सबसेट के उदाहरणों (instances) में ऐसे गुण या संबंध होते हैं, जो केवल सबसेट पर लागू होते हैं।

एसोसिएटिव एंटिटीस (Assocoative Entities) और सामान्यीकरण पदानुक्रमों (Generalization Hierarchies) के बारे में नीचे और अधिक विस्तार से चर्चा की गई है।

2.1.2 रिश्ता (रिलेशनशिप / Relationships) :

एक रिश्ता (Relationship) दो या दो से अधिक एंटिटीस के बीच संबंध (association) को दर्शाता है। एक रिश्ते (Relationship) का एक उदाहरण निम्नानुसार है :

कर्मचारियों को परियोजनाओं (Projects) पर लगाया गया है ।

परियोजनाओं के उपकेंद्र (subtasks) हैं ।

विभाग (Departments) एक या अधिक परियोजनाओं का प्रबंधन करते हैं ।

डिग्री, कनेक्टिविटी, कार्डिनालिटी और अस्तित्व (Degree, Connectivity, Cardinality and existence) के संदर्भ में रिश्तों (Relationship) को वर्गीकृत किया जाता है ।

इन अवधारणाओं पर नीचे चर्चा की जाएगी।

2.1.3 विशेषताएँ / Attributes :

विशेषताएँ उस इकाई **(Entity)** का वर्णन करती हैं**,** जिससे वे संबद्ध हैं । एक विशेषता **(Attribute)** का एक विशेष उदाहरण **(instance),** एक मान (**value**) है। उदाहरण के लिए**,** "किशोर" एट्रिब्यूट "नाम" का एक मान है। एक एट्रिब्यूट का डोमेन (**Domain**) उन सभी संभावित मानों (**Values**) का संग्रह है**,** जो एक एट्रिब्यूट की हो सकती है। **Name** का डोमेन एक वर्ण स्ट्रिंग (**Character String**) है ।

ऐट्रिब्यूट्स को पहचानकर्ता (**Identifiers**) या वर्णनकर्ता **(Descriptors)** के रूप में वर्गीकृत किया जा सकता है । पहचानकर्ता (**Identifiers**)**,** जिन्हें आमतौर पर कुंजी **(Keys)** कहा जाता है**,** विशिष्ट रूप से (**uniquely**) एक एंटिटी के एक उदाहरण (आवृत्ति / **instance**) की पहचान करते हैं। एक वर्णनकर्ता **(Descriptors)** एक एंटिटी उदाहरण (**instance**) की एक गैर-विशिष्ट (**non-unique**) विशेषता का वर्णन करता है।

2.2 रिश्तों का वर्गीकरण (Classifying Relationships) :

रिश्तों (Relationships) को उनकी डिग्री (Degree), कनेक्टिविटी (Connectivity), कार्डिनैलिटी (Cardinality), दिशा (Direction), प्रकार (Type) और अस्तित्व (Existence) द्वारा वर्गीकृत किया जाता है। सभी मॉडलिंग पद्धतियां इन सभी वर्गीकरणों का उपयोग नहीं करती हैं।

2.2.1 रिलेशनशिप की डिग्री (Degree of a Relationship) :

एक रिश्ते (रिलेशनशिप) की डिग्री रिलेशनशिप से जुड़ी एंटिटीस की संख्या है। n-ary रिलेशनशिप, डिग्री n के लिए सामान्य रूप है। बाइनरी (Binary) और टर्नरी (Ternary) विशेष रिलेशनशिप हैं, जहां डिग्री क्रमशः 2, और 3 है।

बाइनरी रिलेशनशिप, दो एंटिटीस के बीच संबंध (association) वास्तविक दुनिया में सबसे आम प्रकार है। एक पुनरावर्ती (recursive) द्विआधारी रिश्ता (बाइनरी रिलेशनशिप) तब होता है जब एक एंटिटी खुद से संबंधित होती है। एक उदाहरण हो सकता है जैसे "कुछ

कर्मचारी दूसरे कर्मचारियों से शादी करते हैं"। यहाँ एंटिटी कर्मचारी का रिलेशनशिप खुद एंटिटी कर्मचारी से ही है ।

एक त्रिगुट रिश्ता (टरनरी रिलेशनशिप) में तीन एंटिटीस शामिल होते हैं और इसका उपयोग तब किया जाता है जब एक बाइनरी रिलेशनशिप अपर्याप्त होता है। कई मॉडलिंग पद्धतियां केवल द्विआधारी रिश्तों (बाइनरी रिलेशनशिप) को मानते हैं। टर्नरी या n-ary रिश्ते, दो या अधिक बाइनरी रिलेशनशिप में विघटित किये जाते हैं।

2.2.2 कनेक्टिविटी और कार्डिनैलिटी (Connectivity and Cardinality):

एक रिश्ते की कनेक्टिविटी (Connectivity of a Relationship), रिश्ते में संबंधित एंटिटी उदाहरणों (instances) की मैपिंग का वर्णन करती है। कनेक्टिविटी का मान (value) "एक" या "कई" होता हैं।

एक रिश्ते की कार्डिनैलिटी (Cardinality of a Relationship), प्रत्येक दो एंटिटीस के लिए संबंधित घटनाओं (occurences) की वास्तविक संख्या होती है।

रिश्तों (Relationship) में मुख्य प्रकार की कनेक्टिविटी (Connectivity) हैं : एक-से-एक, एक-से-कई, और कई-से-कई (one-to-one, one-to-many, many-to-many Relationship)।

2.2.3 रिश्ते की दिशा (Direction of Relationship) :

एक रिश्ते (Relationship) की दिशा (Direction) एक द्विआधारी रिश्ते (Binary Relationship) की उत्पत्ति एंटिटी (Originating Entity) को इंगित करती है। जिस एंटिटी से एक रिश्ता (रिलेशनशिप) उत्पन्न होता है वह पैरेंट एंटिटी (Parent Entity) होती है तथा वह एंटिटी जहां रिलेशनशिप समाप्त हो जाता है , वह चाइल्ड एंटिटी (Child Entity) होती है।

किसी रिलेशनशिप की दिशा (Direction) उसकी कनेक्टिविटी (Connectivity) से तय होती है। एक-से-एक रिश्ते (one-to-one Relationship) में दिशा (Direction), स्वतंत्र एंटिटी (Independent Entity) से एक आश्रित एंटिटी (Dependent Entity) की तरफ होती है। यदि दोनों एंटिटीस स्वतंत्र हैं, तो दिशा कोई भी हो सकती है। एक-से-कई रिश्तों (one-to-many Relationship) में, एक बार आने वाली एंटिटी, पैरेंट एंटिटी होती है। कई-से-कई रिश्तों (many-to-many Relationship) की दिशा कोई भी हो सकती है।

2.3 संबंधपरक कुंजी / (Relational Keys) :

रिश्तों (Relations) में दो तरह की कुंजी (Keys) होती हैं। पहली कुंजी पहचान करने वाली कुंजी होती हैं । इसमें प्राथमिक कुंजी (Primary Key / प्राइमरी की) मुख्य अवधारणा (Concept) है । जबकि दो अन्य कुंजी - सुपर कुंजी (Super Key / सुपर की) और उम्मीदवार कुंजी (Candidate Key / कैंडिडेट की) - संबंधित अवधारणाएं (Related Concepts) हैं। दूसरी तरह की कुंजी विदेशी कुंजी (Foreign Key / फॉरेन की) है।

2.3.1 पहचान की / Identity Keys :

(i) सुपर की / (Super Key)

एक "सुपर की" उन ऐट्रिब्यूट्स का एक समूह है, जिनके मान (values) का उपयोग किसी संबंध (relation) के भीतर एक टपल (tuple) को विशिष्ट रूप से पहचानने के लिए (uniquely identify) किया जा सकता है। एक संबंध (relation) में एक से अधिक सुपर की हो सकती हैं, लेकिन इसमें हमेशा कम से कम एक सुपर की ऐसी होती है जिसमे सभी ऐट्रिब्यूट्स का एक सेट हो जो संबंध (relation) बनाते हैं।

(ii) उम्मीदवार कुंजी (Candidate Key / कैंडिडेट की) :

एक कैंडिडेट की ऐसी एक सुपर कुंजी है जो न्यूनतम है । अर्थात्, इसका कोई उचित उपसमूह (Proper subset) नहीं है, जो स्वयं एक सुपर की हो । एक रिलेशन में एक से अधिक कैंडिडेट की हो सकती है, और अलग-अलग कैंडिडेट की में ऐट्रिब्यूट्स की संख्या अलग-अलग हो सकती हैं। दूसरे शब्दों में आपको, सबसे कम ऐट्रिब्यूट्स वाली सुपर कुंजी ही कैंडिडेट की है , इस प्रकार से 'न्यूनतम' की व्याख्या नहीं करनी चाहिए ।

एक कैंडिडेट की में दो गुण होते हैं:

(a) R के प्रत्येक टपल (tuple) में, K के मान विशिष्ट रूप से उस टूपल की पहचान (uniquely identify) करते हैं : **(विशिष्टता / Uniqueness)**

(b) K के किसी भी प्रॉपर सबसेट (Proper Subset) में विशिष्टता गुण (uniqueness property) ना हो : **(Irreducibility)**

R : Relation और K : key है ।

(iii) प्राथमिक कुंजी (Primary Key) :

एक रिलेशन (relation) की प्राथमिक कुंजी (Primary Key) एक उम्मीदवार कुंजी (Candidate Key) है जिसे विशेष रूप से रिलेशन के लिए कुंजी (Key) चुना जाता है। दूसरे शब्दों में, यह एक कुंजी का चुनाव है, और केवल एक उम्मीदवार कुंजी (Candidate Key) हो सकती है जिसे प्राथमिक कुंजी (Primary Key) के लिए मान्य किया गया हो ।

पहचान कुंजीयों के बीच संबंध :

कुंजीयों के बीच संबंध:

सुपर की (Super Key) ⊇ उम्मीदवार कुंजी (Candidate Key) ⊇ प्राथमिक कुंजी (Primary Key)

2.3.2 विदेशी कुंजी (Foreign Key) :

एक संबंध (relation) के ऐसे एट्रिब्यूटस, जो किसी अन्य संबंध (relation) की एक उम्मीदवार कुंजी (Candidate Key) से मेल खाते है, फॉरेन की कहलाते हैं । एक संबंध (relation) में कई विदेशी कुंजी (फॉरेन की / Foreign Key) हो सकती हैं, ये कुंजियाँ विभिन्न संबंधों (relations) से जुड़ी हो सकती हैं। विदेशी कुंजियाँ (Foreign Key) उपयोगकर्ताओं (users) को एक संबंध (relation) की जानकारी को दूसरे संबंध (relation) की जानकारी से लिंक करने की अनुमति देती हैं। फॉरेन की के बिना, एक डेटाबेस असंबंधित तालिकाओं (tables) का मात्र एक संग्रह होगा।

3. रेफरेंसियल इंटीग्रटी क्या है? (What is Referential Integrity ?)

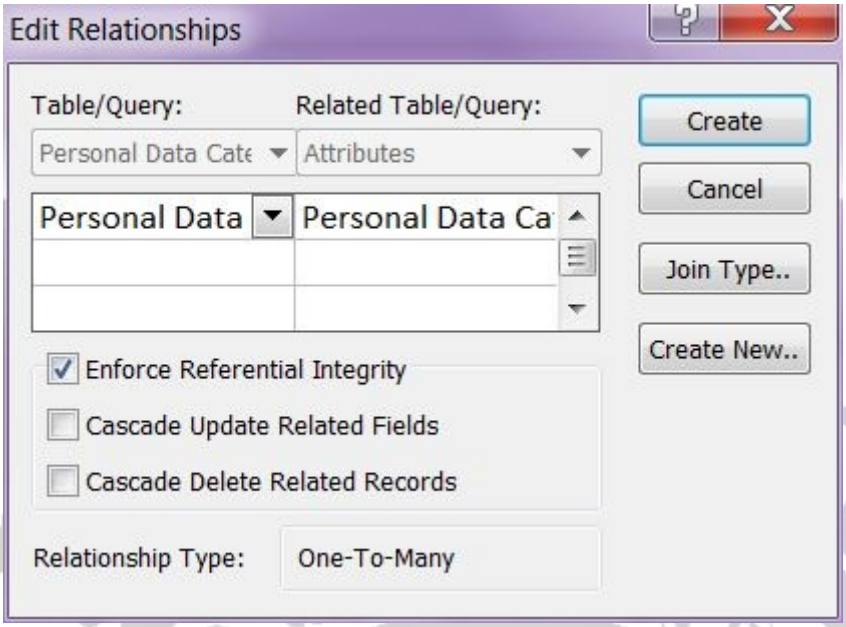
जब आप डेटाबेस डिज़ाइन करते हैं, तो आप डेटा रिडंडेंसी को कम करने के लिए अपनी डेटाबेस जानकारी को कई विषय-आधारित टेबल्स में विभाजित करते हैं। फिर आप सामान्य टेबल्स को संबंधित टेबल्स में रखकर डेटा को एक साथ लाने के लिए एक तरीका प्रदान करते हैं। उदाहरण के लिए, **one-to-many relationship** का प्रतिनिधित्व करने के लिए आप “एक” टेबल से **Primary key** लेते हैं और इसे “**Many”** टेबल में अतिरिक्त फ़ील्ड के रूप में जोड़ते हैं। डेटा को एक साथ वापस लाने के लिए, एक्सेस ” **Many”** टेबल में मान लेता है और “एक” टेबल में संबंधित मान को देखता है। इस तरह ” **Many”** टेबल में मान “एक” टेबल में संबंधित मानों को संदर्भित करते हैं।

टेबल रिलेशनशिप को **Referential Integrity** के मानकों का पालन करना चाहिए, नियमों का एक सेट जो नियंत्रित करता है कि आप संबंधित टेबल्स के बीच डेटा को कैसे हटा या संशोधित कर सकते हैं। टेबल रिलेशनशिप में **Referential Integrity** उपयोगकर्ताओं को गलती से संबंधित डेटा को हटाने या बदलने से रोकती है। आप **Referential Integrity** लागू कर सकते हैं जब: सामान्य फ़ील्ड प्राथमिक टेबल की **Primary key** है; संबंधित फ़ील्ड्स में एक ही प्रारूप है; या दोनों टेबल एक ही डेटाबेस से संबंधित हैं। यदि प्राथमिक टेबल में कर्मचारियों और संबंधित टेबल्स की एक सूची होती है तो उन कर्मचारियों के बारे में अतिरिक्त जानकारी होती है, और एक कर्मचारी छोड़ देता है, उसका रिकॉर्ड प्राथमिक टेबल से हटा दिया जाता है। उनके रिकॉर्ड सभी संबंधित टेबल्स में भी हटा दिए जाने चाहिए। एक्सेस आपको संबंधित डेटा को बदलने या हटाने की अनुमति देता है, लेकिन केवल तभी जब इन परिवर्तनों को संबंधित टेबल्स की श्रृंखला के माध्यम से कैस्केड किया जाता है। आप कैस्केड अपडेट संबंधित फ़ील्ड्स और कैस्केड को रिलेशनशिप कॉन्फ़िगरेशन डायलॉग बॉक्स में संबंधित रिकॉर्ड्स चेक बॉक्स हटाकर इसे कर सकते हैं।

3.1 एमएस एक्सेस में रेफरेंशियल इंटेग्रिटी नियम (Referential Integrity Rules in MS Access)

Relationship Window में Referential Integrity का Option होता हैं इसका अर्थ हैं कि यदि दो टेबल के मध्य **Referential Integrity rule** स्थापित हैं तो प्रथम टेबल में यदि किसी रिकॉर्ड में सुधार या अपडेट करते हैं तो इससे संबंधित टेबल में स्वत: ही संशोधन हो जाता हैं इसी प्रकार यदि प्रथम टेबल में से किसी रिकॉर्ड को **delete** करते हैं तो इससे संबंधित टेबल में से वह रिकॉर्ड **delete** हो जाता हैं। टेबल की **Relationship Create** करते समय **Edit Relationship Window** में **Referential Integrity** के तीन **Option** होते हैं।

a. Enforce Referential Integrity
b. Cascade Update Related Fields
c. Cascade Delete Related Fields



(a) Cascade Update Related Fields :

यदि **Enforce Referential Integrity** के साथ **Cascade Update Related Fields Check Box** को सेलेक्ट करते है तो टेबल के किसी रिकॉर्ड को **Update** करने पर इससे संबंधित टेबल में उस रिकॉर्ड से संबंधित **Information** स्वंयं **Update** हो जाएगी।

(b) Cascade Delete Related Fields :

इसी प्रकार यदि **Cascade Delete Related Check Box** को सेलेक्ट करते हैं तो टेबल में किसी रिकॉर्ड को **Delete** करने पर इससे जुड़ी हुई टेबल में से रिकॉर्ड स्वयं ही **Delete** हो जाएगा।

(c) Enforce Referential Integrity :

Enforce Referential Rule को **Set** करने के लिए **Enforce Referential Integrity Check Box** पर **Click** करते हैं इस के साथ ही अपनी आवश्यकतानुसार **Cascade Update Related Fields** एवं **Cascade Delete Related Fields** पर **Click** करके **Create Button** पर **Click** करते हैं।