



महाराष्ट्र राज्य तंत्रशिक्षण मंडळ, मुंबई
(स्वायत्त्व) (ISO 9001:2015) (ISO/IEC 27001:2013)

अभियांत्रिकी आणि तंत्रज्ञान पदविका

शिक्षण पुस्तिका
(Learning Material)

DATABASE MANAGEMENT SYSTEM

313302

K-Scheme

डाटाबेस मॅनेजमेंट सिस्टिम

संगणक अभियांत्रिकी गट
(के-स्कीम)

मराठी-इंग्रजी (द्विभाषिक माध्यम)
(अभियांत्रिकी व तंत्रज्ञानातील तिसरे सत्र पदविका)

शिक्षण पुस्तिका

(Learning Material)

डाटाबेस मॅनेजमेंट सिस्टिम

Database Management System

(313302)

संगणक अभियांत्रिकी गट

(के-स्कीम)

मराठी-इंग्रजी (द्विभाषिक) माध्यम

(अभियांत्रिकी व तंत्रज्ञानातील तिसरे सत्र पदविका)



महाराष्ट्र राज्य तंत्रशिक्षण मंडळ, मुंबई

(स्वायत्त) (ISO 9001:2015) (ISO/IEC 27001:2013)

डाटाबेस मॅनेजमेंट सिस्टिम
Database Management System
(313302)

मार्गदर्शक

प्रा. विजय नामदेवराव कुकरे
विभागप्रमुख, संगणक अभियांत्रिकी

प्रकल्प समन्वयक

प्रा. विजय तुकाराम पाटील
विभागप्रमुख, संगणक अभियांत्रिकी

संकलक

प्रा. मनीषा पोखरकर
अधिव्याख्याता, संगणक अभियांत्रिकी
प्रा. प्रेरणा जळगावकर
अधिव्याख्याता, माहिती तंत्रज्ञान
प्रा. समिधा चव्हाण
अधिव्याख्याता, माहिती तंत्रज्ञान
प्रा. पूनम पवार
अधिव्याख्याता, संगणक अभियांत्रिकी
प्रा. प्रदिप शिर्के
अधिव्याख्याता, संगणक अभियांत्रिकी



महाराष्ट्र राज्य तंत्र शिक्षण मंडळ

(स्वायत्त) (ISO: ९००१:२०१५) (ISO/IES: २७००१-२०१३)

शासकीय तंत्रनिकेतन इमारत, चौथा मजला, ४९, खेरवाडी, बांद्रा (पूर्व), मुंबई - ४०० ०५१.

दूरध्वनी क्र.: ०२२-६२५४२१७० / १६१

Email : director@msbte.com

Web : www.msbte.org.in



प्रास्ताविक

महाराष्ट्र राज्यातील पदविका स्तरावरील तंत्रशिक्षणामध्ये विद्यार्थ्यांचे रोजगार कौशल्य विकसित करून विद्यार्थ्यांचा सर्वांगीण विकास घडवून आणण्याकरिता महाराष्ट्र राज्य तंत्रशिक्षण मंडळ कटिबद्ध आहे. उद्योगधंद्यातील बदलत्या तंत्रज्ञानाशी संबंधित गरजा लक्षात घेऊन महाराष्ट्र राज्य तंत्र शिक्षण मंडळाकडून पदविका अभ्यासक्रम वेळोवेळी अद्यावत करण्यात येतो. अभियांत्रिकी पदविका अभ्यासक्रम शिकत असतांना संकल्पनात्मक ज्ञान, सुसंगत संदर्भ, प्रश्न विचारणे, विश्वसनिय पुरावे, कारणमीमांसा आणि सुस्पष्ट निकष यांचा वापर करून अर्थाची उकल करण्याची, विश्लेषण व मूल्यमापन करण्याची तसेच तर्काने अनुमान काढण्याची क्षमता म्हणजेच चिकित्सक विचार विद्यार्थ्यांमध्ये अधिक दृढ होतील असा मला विश्वास आहे. जेव्हा विद्यार्थी ज्ञान मिळवण्याच्या माध्यमाशी पूर्णपणे परिचित आणि सोयीस्कर असतात, तेव्हा त्यांच्यासाठी वर्गातील चर्चेत भाग घेणे सोपे होते, संकल्पनात्मक व सैद्धांतिक बाबींचे आकलन परिपूर्ण होते, संज्ञानात्मक क्षमता सुधारते आणि त्यांचा आत्मविश्वास देखील वाढतो या सर्व गोष्टींचा विचार करून मंडळाकडून शैक्षणिक सामुग्रीची निर्मिती करण्यात आलेली आहे. भारत देश हा खेड्यापाड्यातून विकसित झालेला देश असून ग्रामीण भागातील विद्यार्थ्यांना तांत्रिक शिक्षण घेतांना भाषेचा अडसर न येता तांत्रिक बाबींचा आशय समजून घेणे शक्य होईल या दृष्टिकोनातून महाराष्ट्र राज्य तंत्र शिक्षण मंडळाने पदविका स्तरावरील तांत्रिक शिक्षणाकरिता विद्यार्थ्यांना मराठी-इंग्रजी द्विभाषिक माध्यमाचा पर्याय शैक्षणिक वर्ष २०२१-२२ पासून उपलब्ध करून दिलेला आहे.

राष्ट्रीय शैक्षणिक धोरण-२०२० प्रादेशिक भाषेतील शिक्षणास प्रोत्साहन देते, ज्यामुळे विद्यार्थ्यांना तांत्रिक अभ्यासक्रमासाठी प्रादेशिक भाषांतुन शिक्षणाचे माध्यम निवडता येते. सदर धोरणामुळे प्रादेशिक भाषांमध्ये तांत्रिक सामग्री आणि अभ्यास सामग्रीचा विकास आणि भाषांतर निर्माण करण्याची आवश्यकता आहे. त्यास अनुसरून मंडळाने मराठी-इंग्रजी द्विभाषिक माध्यमाचा पर्याय द्वितीय व तृतीय वर्षाकरिताही उपलब्ध करून देण्यात आला आहे. तसेच त्याकरिताची शैक्षणिक सामग्रीही संबंधीत भागधारकरांना उपलब्ध करून देण्यात येत आहे.

पदविका स्तरावरील तंत्रशिक्षण अधिक दर्जेदार करण्यासाठी महाराष्ट्रातील अनुभवी व तज्ञ अध्यापकांनी व्यवहारिक मराठी भाषा व इंग्रजी भाषेतील तांत्रिक शब्दावली यांचा वापर करून मराठी - इंग्रजी भाषेचा सुवर्णमध्य साधण्याचा प्रयत्न केलेला आहे. मंडळाच्या स्तरावर गठीत सुकाणू समितीमार्फत सदर शैक्षणिक सामुग्रीचा दर्जा, तसेच इतर बाबींची तपासणी करण्यात आलेली आहे. त्यामुळे सदर शैक्षणिक सामुग्री अधिक सम्पन्न झालेली असून विद्यार्थी त्यांच्या व्यक्तिमत्त्वाचा सुसंवादी आणि सर्वांगीण विकास साधतील. परिणामतः विश्वस्तरीय मनुष्यबळाच्या गरजा पूर्ण करण्यात महाराष्ट्र राज्य अग्रेसर राहील व पर्यायाने राष्ट्रनिर्मिती करीता निश्चितच हातभार लागेल, असा मला विश्वास आहे.

अभियांत्रिकी पदविका अभ्यासक्रमातील प्रमुख विषयांची मराठी-इंग्रजी द्विभाषिक शैक्षणिक सामुग्री बनविण्यासाठी अध्यापक व सुकाणू समितीचे सदस्य यांनी दर्शविलेले समर्पण व वचनबद्धता कौतुकास पात्र आहे, या सर्वांचे मी मनः पूवक अभिनंदन करतो !



(प्रमोद नाईक)

संचालक

म. रा. तंत्र शिक्षण मंडळ, मुंबई.

अनुक्रमणिका

अ. नु.	युनिटचे नाव	पृष्ठ क्रमांक
1	इंट्रोडक्शन टू डाटाबेस (Introduction to Database System)	1
2	रिलेशनल डाटा मॉडेल (Relational Data Model)	14
3	इंटरॅक्टिव्ह SQL अँड परफॉर्मन्स ट्युनिंग (Interactive SQL and Performance Tuning)	26
4	PL / SQL प्रोग्रामिंग (PL/SQL Programming)	43
5	डाटाबेस ऍडमिनिस्ट्रेशन (Database Administration)	83

युनिट -1

इंट्रोडक्शन टू डाटाबेस सिस्टम

(Introduction to Database System)

विषय निष्पत्ती (Course Outcome): डाटाबेस व्यवस्थापन प्रणालीची संकल्पना स्पष्ट करा

घटक निष्पत्ती (Theory Learning Outcome):

1. दिलेल्या डाटाबेस सिस्टमचे स्पष्टीकरण द्या
2. DBMS ची एकूण रचना स्पष्ट करा
3. डाटाबेसच्या आर्किटेक्चरचे वर्णन करा

1.1 डाटाबेस संकल्पना (Database Concept):

1.1.1 डाटा (Data): डाटा म्हणजे संगणकासह वापरण्यासाठी योग्य फॉर्ममध्ये असलेली कोणतीही गोष्ट. डाटा म्हणजे अशी माहिती जी संगणकात साठवून त्यावर प्रक्रिया केली जाऊ शकते. डाटामध्ये तथ्ये, मजकूर, ग्राफिक्स, प्रतिमा, ऑडिओ आणि व्हिडिओ विभाग हे पर्यावरणाचे महत्त्व वापरणारे आहेत. डाटाबेस हा संबंधित डाटाचा एक संघटित संच आहे. डाटा वापरकर्त्याद्वारे सहजपणे संग्रहित, हाताळणी आणि पुनर्प्राप्त करण्यासाठी संरचित आहे.

उदाहरणार्थ : विद्यार्थी डाटा जसे विद्यार्थ्याचे नाव, पत्ता, जन्मतारीख इत्यादी.

1.1.2 डाटाबेस (Database) : डाटाबेस हा माहितीचा संग्रह आहे जो सहजपणे ऍक्सेस (access), व्यवस्थापित (manage) आणि अद्यतनित (update) केला जाऊ शकतो. डाटाबेस हा डाटाबेस मॅनेजमेंट सिस्टम (Database Management System) द्वारे नियंत्रित केला जातो. डाटाबेस हा डाटाचा संग्रह आहे आणि एखाद्या विशिष्ट एंटरप्राइझ किंवा संस्थेची उपयुक्त माहिती संग्रहित करण्यासाठी वापरला जातो. डाटाबेस हा प्रमाणित स्वरूपात संग्रहित केलेल्या संबंधित डाटाचा संग्रह आहे, जो एकाधिक वापरकर्त्याद्वारे सामायिक केला जाऊ शकतो. एक्सेल सारण्यांप्रमाणे, डाटाबेस टेबलमध्ये स्तंभ आणि पंक्ती असतात. प्रत्येक स्तंभामध्ये भिन्न प्रकारची विशेषता असते आणि प्रत्येक पंक्ती एका रेकॉर्डशी संबंधित असते.

उदाहरणार्थ : शाळेच्या डाटाबेससाठी (School Database) विद्यार्थी, शिक्षक, वर्ग, परीक्षा ह्या संदर्भातील डाटा संकलित केला जातो.

1.1.3 डाटाबेस मॅनेजमेंट सिस्टम (Database Management System): डाटाबेस मॅनेजमेंट सिस्टम (DBMS) ही एक सॉफ्टवेअर मॅनेजमेंट सिस्टम आहे जी डाटा स्टोरेज (Data storage), डाटा ऍक्सेस (Data access) आणि डाटाची सुरक्षा (Data security) देण्यासाठी बनवली आहे.

डाटाबेस मॅनेजमेंट सिस्टीम (DBMS) हा परस्परसंबंधित डाटा आणि त्या डाटामध्ये प्रवेश करण्यासाठी प्रोग्रामचा संच आहे. डाटाचे संकलन, सामान्यतः डाटाबेस म्हणून ओळखले जाते, त्यात एका विशिष्ट एंटरप्राइझबद्दल माहिती असते.

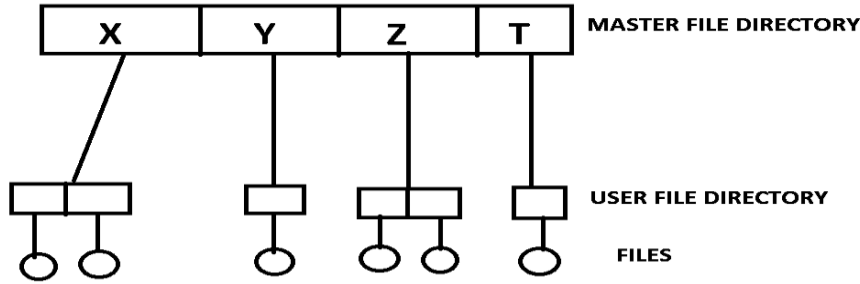
- डाटाबेस व्यवस्थापन प्रणाली हा सॉफ्टवेअर प्रोग्रामचा एक संच आहे जो वापरकर्त्यांना डाटाबेस फायलींमध्ये डाटा तयार करण्यास, संपादित करण्यास आणि अद्यतनित करण्यास आणि त्या डाटाबेस फायलींमधून डाटा संचयित आणि पुनर्प्राप्त करण्यास अनुमती देतो. डाटाबेसमधील डाटा डीबीएमएस वापरून जोडला, हटवला, बदलला, क्रमवारी लावला किंवा शोधला जाऊ शकतो.
- सिस्टमच्या वापरकर्त्यांना डाटाबेसमधील डाटामध्ये फेरफार करण्यासाठी किंवा डाटाबेस संरचना स्वतःच व्यवस्थापन करण्यासाठी अशा प्रणालीवर अनेक प्रकारची ऑपरेशन्स करण्याची सुविधा दिली जाते. डाटाबेस मॅनेजमेंट सिस्टम (DBMS) ची त्यांच्या डाटा संरचना किंवा प्रकारांनुसार वर्गीकरण केले जाते.

उदाहरणार्थ : Oracle ,MySQL, SQLite, IBM Db2 ही काही डाटाबेस मॅनेजमेंट सिस्टिम्स ची नावे आहेत.

1.1.4 फाइल सिस्टम विरुद्ध डीबीएमएस (File System versus DBMS) :

फाइल सिस्टम हा मुळात हार्ड डिस्कसारख्या स्टोरेज माध्यमात फाइल्सची व्यवस्था करण्याचा एक मार्ग आहे. फाइल सिस्टम फाइल्स व्यवस्थित करते आणि आवश्यकतेनुसार फाइल्स पुनर्प्राप्त करण्यात मदत करते. फाइल सिस्टममध्ये वेगवेगळ्या फाइल्स असतात ज्या डिरेक्टरीमध्ये गटबद्ध केल्या जातात. डिरेक्टरीमध्ये इतर फोल्डर्स आणि फाइल्स असतात. फाइल सिस्टीम व्यवस्थापन, फाइलचे नाव देणे, प्रवेश नियम देणे इत्यादी मूलभूत ऑपरेशन्स करते.

उदाहरण: NTFS (नवीन तंत्रज्ञान फाइल Management, EXT-विस्तारित फाइल Management).



आकृती 1.1 फाइल सिस्टम विरुद्ध डीबीएमएस (File System versus DBMS)

1.1.5 डाटाबेस व्यवस्थापन (DBMS): डाटाबेस मॅनेजमेंट सिस्टम हे मुळात सॉफ्टवेअर आहे जे संबंधित डाटाचे संकलन व्यवस्थापित करते. डाटा संग्रहित करण्यासाठी आणि आवश्यकतेनुसार डाटा प्रभावीपणे पुनर्प्राप्त करण्यासाठी याचा वापर केला जातो. हे अनधिकृत प्रवेशापासून डाटाचे संरक्षण करण्यासाठी योग्य सुरक्षा उपाय देखील प्रदान करते.

1.1.6 फाइल सिस्टम आणि DBMS मधील फरक :

मूलभूत	फाइल सिस्टीम (File System)	डाटाबेस व्यवस्थापन (DBMS)
रचना (Structure)	फाइल सिस्टीम ही संगणकातील स्टोरेज माध्यमात फाइल्सची मांडणी करण्याचा एक मार्ग आहे.	DBMS हे डाटाबेस व्यवस्थापित करण्यासाठी सॉफ्टवेअर आहे.
डाटा रिडंडंसी (Data Redundancy)	फाइल सिस्टममध्ये अनावश्यक डाटा असू शकतो. (File system can have redundant data)	DBMS मध्ये कोणताही अनावश्यक डाटा नाही. (No redundant data in DBMS)
बॅकअप आणि पुनर्प्राप्ती (Backup and Recovery)	डाटा हरवल्यास बॅकअप आणि पुनर्प्राप्तीसाठी ते अंगभूत यंत्रणा प्रदान करत नाही. (No Backup and Recovery mechanism provided)	डाटा हरवला असला तरीही बॅकअप आणि पुनर्प्राप्तीसाठी हे साधने प्रदान करते. (Backup and Recovery mechanism is provided)
क्वेरी प्रक्रिया (Query Processing)	फाइल सिस्टममध्ये कोणतीही कार्यक्षम क्वेरी प्रक्रिया नाही. (There is no efficient query processing)	DBMS मध्ये कार्यक्षम क्वेरी प्रक्रिया आहे. (There is efficient query processing)

मूलभूत	फाईल सिस्टीम (File System)	डाटाबेस व्यवस्थापन (DBMS)
गुंतागुंत (Complexity)	DBMS च्या तुलनेत कमी क्लिष्ट आहे. (It is less complex)	फाईल सिस्टीम च्या तुलनेत हाताळणीत अधिक जटिलता आहे. (It has more complexity)
सुरक्षा मर्यादा (Security Constraints)	DBMS च्या तुलनेत फाईल सिस्टीम कमी सुरक्षा प्रदान करतात. (File systems provide less security)	फाईल सिस्टीम च्या तुलनेत DBMS मध्ये अधिक सुरक्षा यंत्रणा आहेत. (DBMS has more security)
खर्च (Cost)	हे DBMS पेक्षा कमी खर्चिक आहे. (Cost is less than DBMS)	फाईल सिस्टीम पेक्षा त्याची किंमत तुलनेने जास्त आहे. (Cost is more than File system)

1.1.7 डीबीएमएसचे ॲप्लिकेशन (Applications of Database Management System):

- 1. रेल्वे आरक्षण (Railway Reservation):** रेल्वे मार्ग आरक्षण फ्रेमवर्कमध्ये, तिकीट भेटीचे रेकॉर्ड किंवा माहिती, ट्रेनचे स्वरूप आणि उड्डाण यांची माहिती संग्रहित करण्यासाठी माहिती बेस आवश्यक आहे. याशिवाय, ट्रेन्स उशीरा आल्यास, माहिती बेस अपडेटद्वारे व्यक्ती त्याच्याशी परिचित होतात.
- 2. ग्रंथालय व्यवस्थापन (Library Management):** ग्रंथालयात अनेक पुस्तके आहेत त्यामुळे; सापेक्ष पुस्तकांच्या संख्येची नोंद रजिस्टर किंवा डुप्लिकेटमध्ये संग्रहित करणे कठीण आहे. या ओळींसह, डाटाबेस ॲडमिनिस्ट्रेशन फ्रेमवर्क (DBMS) चा वापर पुस्तकाचे नाव, अंकाची तारीख, पुस्तकाची प्रवेशयोग्यता आणि लेखकासह ओळखला जाणारा सर्व डाटा ठेवण्यासाठी केला जातो.
- 3. बँकिंग (Banking):** माहिती बेसमध्ये क्लायंटचा एक्सचेंज डाटा संग्रहित करण्यासाठी कार्यकारी फ्रेमवर्कचा डाटाबेस वापरला जातो.
- 4. शिक्षण क्षेत्र (Education):** सध्या, अनेक शाळा आणि महाविद्यालयांद्वारे मूल्यांकन ऑनलाइन केले जाते. ते डाटाबेस ॲडमिनिस्ट्रेशन फ्रेमवर्क (DBMS) द्वारे सर्व मूल्यांकन माहिती हाताळतात. त्या अभ्यासाच्या नावनोंदणी बारकावे, ग्रेड, अभ्यासक्रम, खर्च, सहभाग, परिणाम, आणि असे असूनही सर्व डाटा माहिती बेस मध्ये दूर ठेवले आहे.
- 5. क्रेडिट कार्ड एक्सचेंज (Credit Card Exchange):** डाटाबेस मॅनेजमेंट फ्रेमवर्कचा उपयोग चार्ज कार्ड्सवर खरेदी करण्यासाठी आणि महिन्यापासून महिन्याच्या घोषणेसाठी केला जातो.
- 6. सोशल मीडिया साइट्स (Social Media):** आम्ही सर्व ऑनलाइन मीडिया साइट्सचा वापर साथीदारांशी जोडण्यासाठी आणि जगाला आमचे दृष्टीकोन देण्यासाठी करतो. दररोज, बरेच लोक या ऑनलाइन मीडिया खात्यांचा पाठपुरावा करतात जसे की Pinterest, Facebook, Twitter आणि Google व्यतिरिक्त. डाटाबेस ॲडमिनिस्ट्रेशन फ्रेमवर्कचा वापर करून, क्लायंटचा सर्व डाटा माहिती बेसमध्ये ठेवला जातो आणि आम्ही इतरांशी संवाद साधण्यास तयार होतो.

1.1.8 डाटा ॲब्सट्रॅक्शन (Data Abstraction):

डाटाबेस सिस्टममध्ये गुंतागुंतीच्या डाटा स्ट्रक्चर्स आणि संबंध असतात. डेव्हलपर वापरकर्त्यापासून कॉम्प्लेक्स डाटा दूर ठेवतात आणि गुंतागुंत दूर करतात जेणेकरून वापरकर्ता डाटाबेसमधील डाटामध्ये आरामात प्रवेश करू शकेल आणि फक्त त्यांना हवा असलेला डाटा ॲक्सेस करू शकेल, जे डाटा ॲब्सट्रॅक्शनच्या मदतीने केले जाते.

डाटा ॲब्सट्रॅक्शनचा मुख्य उद्देश इररेलेवंट डाटा लपवणे आणि डाटाचे ॲब्सट्रॅक्ट व्ह्यू प्रदान करणे आहे. डाटा ॲब्सट्रॅक्शनच्या मदतीने, डेव्हलपर वापरकर्त्याकडून अप्रासंगिक डाटा लपवतात आणि त्यांना संबंधित डाटा

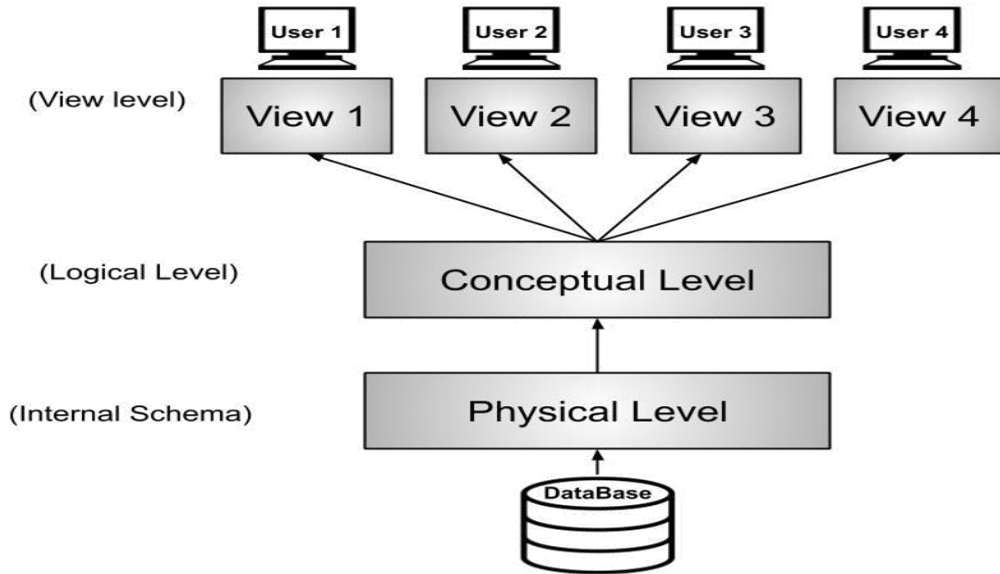
देतात.. असे केल्याने, वापरकर्ते कोणत्याही अडचणीशिवाय डाटा ऍक्सेस करू शकतात आणि सिस्टम देखील कार्यक्षमतेने कार्य करेल.

Database User पासून डाटाबेसचे कॉम्प्लेक्स डिटेल लपवणे ह्याला डाटा अॅब्सट्रॅक्शन (Data Abstraction) असे म्हणतात.

1.1.8.1 डीबीएमएसमधील डाटा अॅब्सट्रॅक्शन्सचे स्तर (Levels of Data Abstraction):

डीबीएमएसमध्ये, डाटा अॅब्सट्रॅक्शनचे तीन स्तर आहेत, जे खालीलप्रमाणे आहेत:

1. फिजिकल किंवा इंटरनल लेवल (Physical or Internal Level)
2. लॉजिकल किंवा कन्सेप्युअल लेवल (Logical or Conceptual Level)
3. व्ह्यू किंवा एक्स्टर्नल लेवल (View or External Level)



आकृती 1. 2 डाटा अॅब्सट्रॅक्शन्सचे स्तर (Levels of Data Abstraction)

1. फिजिकल किंवा इंटरनल लेवल (Physical or Internal Level):

डाटा अॅब्सट्रॅक्शनची ही सर्वात खालची लेवल आहे. ही लेवल मेमरीमध्ये डाटा कसा संग्रहित केला जातो ते सांगते. फिजिकल लेवल ला "इंटरनल लेवल" असेही म्हणतात.

2. लॉजिकल किंवा कन्सेप्युअल लेवल (Logical or Conceptual Level):

या स्तरामध्ये डाटाबेसमध्ये टेबलच्या (Table) स्वरूपात साठवलेली माहिती समाविष्ट असते. हे तुलनेने सोप्या संरचनांमध्ये डाटा घटकांमधील संबंध देखील संग्रहित करते. लॉजिकल लेवल ला "कन्सेप्युअल लेवल" असेही म्हणतात.

3. व्ह्यू किंवा एक्स्टर्नल लेवल (View or External Level):

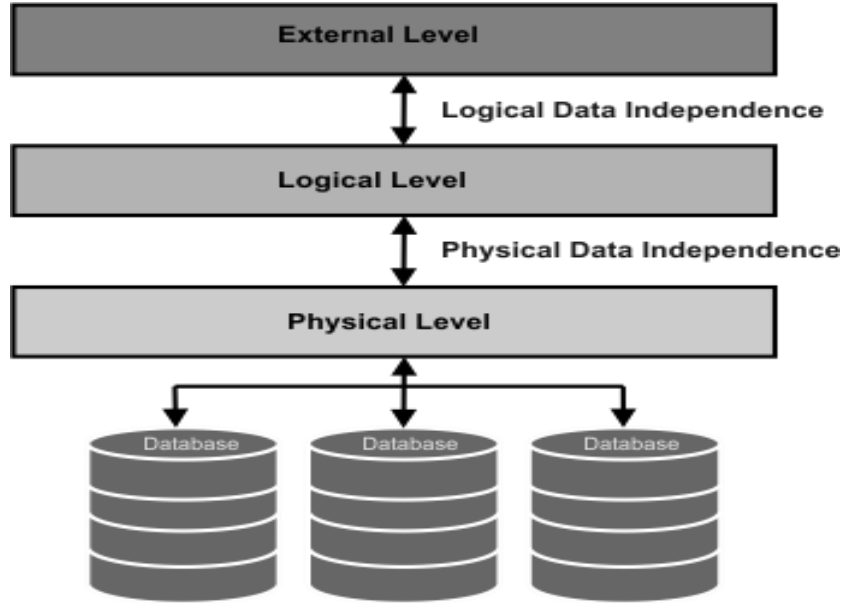
डाटा अॅब्सट्रॅक्शनची ही सर्वात वरची लेवल आहे. वापरकर्त्या द्वारे डाटाबेसचा फक्त काही भाग पाहिला (User can view part of table) जातो. उर्वरित डाटाबेस वापरकर्त्या (User) पासून लपवला जातो. वापरकर्ते (User) पंक्ती आणि स्तंभांच्या (Rows and columns) स्वरूपात डाटा पाहतात. डाटा संग्रहित करण्यासाठी टेबल आणि संबंध वापरले जातात. एकाच डाटाबेसची अनेक दृश्ये (multiple views) अस्तित्वात असू शकतात. व्ह्यू लेवल ला "एक्स्टर्नल लेवल" असेही म्हणतात.

1.1.9 डाटा इंडिपेंडन्स (Data Independence):

पुढील स्तरावरील स्कीमा व्याख्येवर परिणाम न करता एका स्तरावर स्कीमा व्याख्या सुधारण्याची क्षमता याला "डाटा इंडिपेंडन्स" म्हणतात. डाटा इंडिपेंडन्स हे मुख्यत्वे DBMS चे गुणधर्म म्हणून परिभाषित केले जाते जे

तुम्हाला पुढील लेवल मधील स्कीमा बदलण्याची आवश्यकता न ठेवता सिस्टमच्या एका स्तरावर डाटाबेस स्कीमा बदलण्यास मदत करते. ते डाटा वापरणाऱ्या सर्व प्रोग्रामपासून वेगळे ठेवण्यास मदत करते. आपल्याकडे डाटा इंडिपेंडन्सचे दोन स्तर आहेत, जे खालीलप्रमाणे आहेत:

- फिजिकल डाटा इंडिपेंडन्स (Physical Data Independence)
- लॉजिकल डाटा इंडिपेंडन्स (Logical Data Independence)



आकृती 1.3 डाटा इंडिपेंडन्स (Data Independence)

फिजिकल डाटा इंडिपेंडन्स (Physical Data Independence): लॉजिकल स्कीमा आणि व्ह्यू स्कीमा व्याख्येवर परिणाम न करता फिजिकल स्कीमा व्याख्या सुधारण्याची क्षमता याला "फिजिकल डाटा इंडिपेंडन्स" म्हणतात. हे ऑप्टिमायझेशनच्या उद्देशाने केलेल्या संकल्पनात्मक किंवा तार्किक स्कीमामध्ये कोणतेही बदल न करता भौतिक स्कीमा सुधारण्यास सक्षम असण्याच्या वैशिष्ट्याचा संदर्भ देते, उदा., डाटाबेस सिस्टमच्या स्टोरेज आकारात कोणत्याही बदलामुळे डाटाबेसची संकल्पनात्मक रचना प्रभावित होणार नाही. सर्व्हर अनुक्रमिक ते यादृच्छिक प्रवेश फाइल्समध्ये बदलणे हे असेच एक उदाहरण आहे. हे बदल किंवा भौतिक रचनेतील बदलांमध्ये हे समाविष्ट असू शकते:

- नवीन स्टोरेज उपकरणे वापरणे.
- स्टोरेजसाठी वापरल्या जाणाऱ्या डाटा स्ट्रक्चर्समध्ये बदल करणे.
- निर्देशांक बदलणे किंवा पर्यायी फाइल संस्था तंत्रे वापरणे इ.

लॉजिकल डाटा इंडिपेंडन्स (Logical Data Independence): व्ह्यू स्कीमा व्याख्येवर परिणाम न करता लॉजिकल स्कीमा व्याख्या सुधारण्याची क्षमता याला "लॉजिकल डाटा इंडिपेंडन्स" म्हणतात. बाह्य स्कीमा किंवा ऍप्लिकेशन प्रोग्रामला प्रभावित न करता लॉजिकल स्कीमा सुधारण्यात सक्षम होण्याच्या वैशिष्ट्याचा संदर्भ देते. डाटाच्या वैचारिक दृश्यातील कोणत्याही बदलांमुळे डाटाचे वापरकर्ता दृश्य प्रभावित होणार नाही. या बदलांमध्ये विशेषता समाविष्ट करणे किंवा हटवणे, सारणी संरचना घटकांमध्ये बदल करणे किंवा लॉजिकल स्कीमाशी संबंध इत्यादींचा समावेश असू शकतो.

1.1.10 डाटाबेस स्कीमा (Database Schema): डाटाबेस स्कीमा ही डाटाबेसची तार्किक रचना आहे. डाटाबेस स्कीमा एट्रीब्यूटसद्वारे तयार केला जातो आणि या स्केलेटनला स्कीमा असे नाव दिले जाते. स्कीमा टेबल, प्राइमरी की इत्यादीसारख्या तार्किक मर्यादांचा उल्लेख करते. स्कीमा विशेषतांचा डाटा प्रकार दर्शवत नाही.

For Example : Table -Student

RollNo	Name	Percentage	Branch

स्टुडेंट टेबल चा स्कीमा खालील प्रमाणे

Student (RollNo, Name, Percentage, Branch)

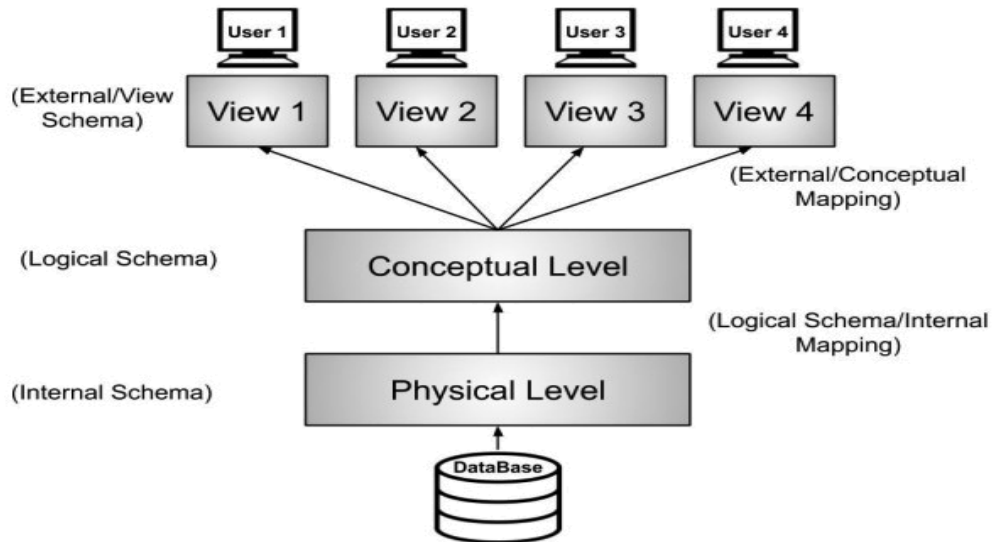
आकृती 1.4 स्टुडेंट टेबल स्कीमा (Student table Schema)

डाटाबेस स्कीमा हे डाटाचे तार्किक प्रतिनिधित्व आहे जे दर्शविते की डाटाबेसमधील डाटा तार्किकरित्या कसा संग्रहित केला जावा. डाटा कसा व्यवस्थित केला जातो आणि सारण्यांमधील संबंध हे दर्शविते. डाटाबेस स्कीमामध्ये सारणी, फील्ड, दृश्ये आणि विविध कीजमधील संबंध असतात जसे की प्राइमरी की (Primary key), फॉरेन की (Foreign key).

डाटाबेस स्कीमाचे प्रकार (Types of Database Schemas):

डाटाबेस स्कीमाचे तीन प्रकार आहेत , जे खालीलप्रमाणे आहेत:

1. फिजिकल स्कीमा (Physical Schema)
2. लॉजिकल स्कीमा (Logical Schema)
3. व्ह्यू स्कीमा (View Schema)



आकृती 1.5 डाटाबेस स्कीमाचे प्रकार (Types of Database Schemas)

1. फिजिकल स्कीमा (Physical Schema): डाटा, फाइल्स, माहिती स्टोरेज सिस्टममध्ये भौतिकरित्या कशी संग्रहित केली जाते ते फिजिकल डाटाबेस स्कीमा परिभाषित करते. डाटाबेसची रचना तयार करण्यासाठी हा वास्तविक कोड किंवा वाक्यरचना आवश्यक आहे, आपण असे म्हणू शकतो की जेव्हा आपण भौतिक स्तरावर डाटाबेस डिझाइन करतो तेव्हा त्याला भौतिक स्कीमा म्हणतात. डाटाबेस प्रशासक वेगवेगळ्या स्टोरेज ब्लॉक्समध्ये डाटा कुठे आणि कसा संग्रहित करायचा ते निवडतो.

2. लॉजिकल स्कीमा (Logical Schema): लॉजिकल स्कीमा सर्व तार्किक मर्यादा परिभाषित करते .जे संग्रहित डाटावर लागू करणे आवश्यक आहे आणि सारण्या, दृश्ये, अस्तित्व संबंध आणि अखंडता मर्यादा यांचे वर्णन देखील करते. तार्किक योजना सारणीच्या स्वरूपात डाटा कसा संग्रहित केला जातो आणि सारणीचे गुणधर्म कसे जोडले जातात याचे वर्णन करते.ER मॉडेलिंग वापरून डाटाच्या घटकांमधील संबंध राखला जातो. तार्किक स्कीमामध्ये समाविष्ट करण्याची गुणवत्ता राखण्यासाठी आणि डाटा अद्यतनित करण्यासाठी भिन्न अखंडता मर्यादा परिभाषित केल्या आहेत.

3. व्ह्यू स्कीमा (View Schema): व्ह्यू स्कीमा जे अंतिम वापरकर्ता आणि डाटाबेस यांच्यातील परस्परसंवाद परिभाषित करण्यास सक्षम आहे. डाटाबेसमधील डाटाच्या संग्रहित यंत्रणेबद्दल जास्त माहिती न घेता वापरकर्ता इंटरफेसच्या मदतीने डाटाबेसशी संवाद साधण्यास सक्षम आहे.

1.1.11 डीबीएमएस मधील कॉडचे नियम (Codd's Rules in DBMS): कॉडचे नियम डॉ. एडगर एफ. कॉड नावाच्या संगणक शास्त्रज्ञाने प्रस्तावित केले आहेत आणि त्यांनी डाटाबेस व्यवस्थापनासाठी रिलेशनल मॉडेलचाही शोध लावला आहे. हे नियम डाटा अखंडता, सातत्य आणि उपयोगिता सुनिश्चित करण्यासाठी बनवले आहेत. नियमांचा हा संच मुळात रिलेशनल डाटाबेस मॅनेजमेंट सिस्टम (RDBMS) ची वैशिष्ट्ये आणि आवश्यकता दर्शवतो. या लेखात आपण कॉडच्या विविध नियमांबद्दल जाणून घेणार आहोत.

डीबीएमएस मधील कॉडचे नियम :

नियम 1: माहिती नियम (Information Rule): सर्व माहिती, मग ती वापरकर्ता माहिती असो किंवा मेटाडाटा, जी डाटाबेसमध्ये संग्रहित केली जाते, ती टेबलच्या सेल्समध्ये मूल्य म्हणून प्रविष्ट केली जाणे आवश्यक आहे. असे म्हटले जाते की डाटाबेसमधील प्रत्येक गोष्ट टेबल लेआउटमध्ये आयोजित केली जाते.

नियम 2: गॅरंटेड एक्सेस नियम (Guaranteed Access Rule): सारणीचे नाव (Table Name), प्राथमिक की (Primary Key) आणि स्तंभ मूल्य (Column Name) यांच्या संयोजनासह प्रत्येक डाटा घटक तार्किकदृष्ट्या प्रवेशयोग्य असण्याची हमी दिली जाते.

नियम 3: सिस्टमतिक ट्रीटमेंट ऑफ नल व्हॅल्यू नियम (Systematic Treatment of NULL Values): डाटाबेसमधील प्रत्येक नल मूल्याला (Null value) योग्य दर्जा दिला जाणे आवश्यक आहे.

नियम 4: एक्टिव्ह ऑनलाइन कॅटलॉग नियम (Active Online Catalog): डाटाबेस कॅटलॉग, ज्यामध्ये डाटाबेसबद्दलचा मेटाडाटा (metadata) असतो, त्याच रिलेशनल डाटाबेस मॅनेजमेंट सिस्टमचा वापर करून संग्रहित आणि एक्सेस करणे आवश्यक आहे.

नियम 5: कॉम्प्रेहेन्सिव्ह डाटा सब लॅंग्वेज नियम (Comprehensive Data Sub-Language Rule): कोणत्याही कार्यक्षम डाटाबेस प्रणालीचा एक महत्वाचा घटक म्हणजे सहजपणे समजण्यायोग्य डाटा मॅनिप्युलेशन लॅंग्वेज (DML) ऑफर करण्याची क्षमता आहे. DML डाटाबेसमधील माहिती संग्रहित करणे, क्वेरी करणे आणि सुधारित करणे सुलभ करते.

नियम 6: व्ह्यू अपडेटिंग नियम (View Updating Rule): सर्व व्ह्यू (views) सिद्धांतिक दृष्ट्या (theoretically) आणि सिस्टिम द्वारा बदलता येतात.

नियम 7: हाय लेवल इन्सर्ट, अपडेट अँड डिलीट नियम (High-Level Insert, Update, and Delete Rule): डाटाबेस सिस्टिम, वापरकर्त्यांना (users) एका क्वेरीद्वारे सहजतेने insert, update and delete ही ऑपरेशन्स करण्याची क्षमता देऊ शकते.

नियम 8: फिजिकल डाटा इंडिपेंडन्स नियम (Physical Data Independence): लॉजिकल स्कीमा आणि व्ह्यू स्कीमा व्याख्येवर परिणाम न करता फिजिकल स्कीमा व्याख्या सुधारण्याची क्षमता याला "फिजिकल डाटा इंडिपेंडन्स" म्हणतात.

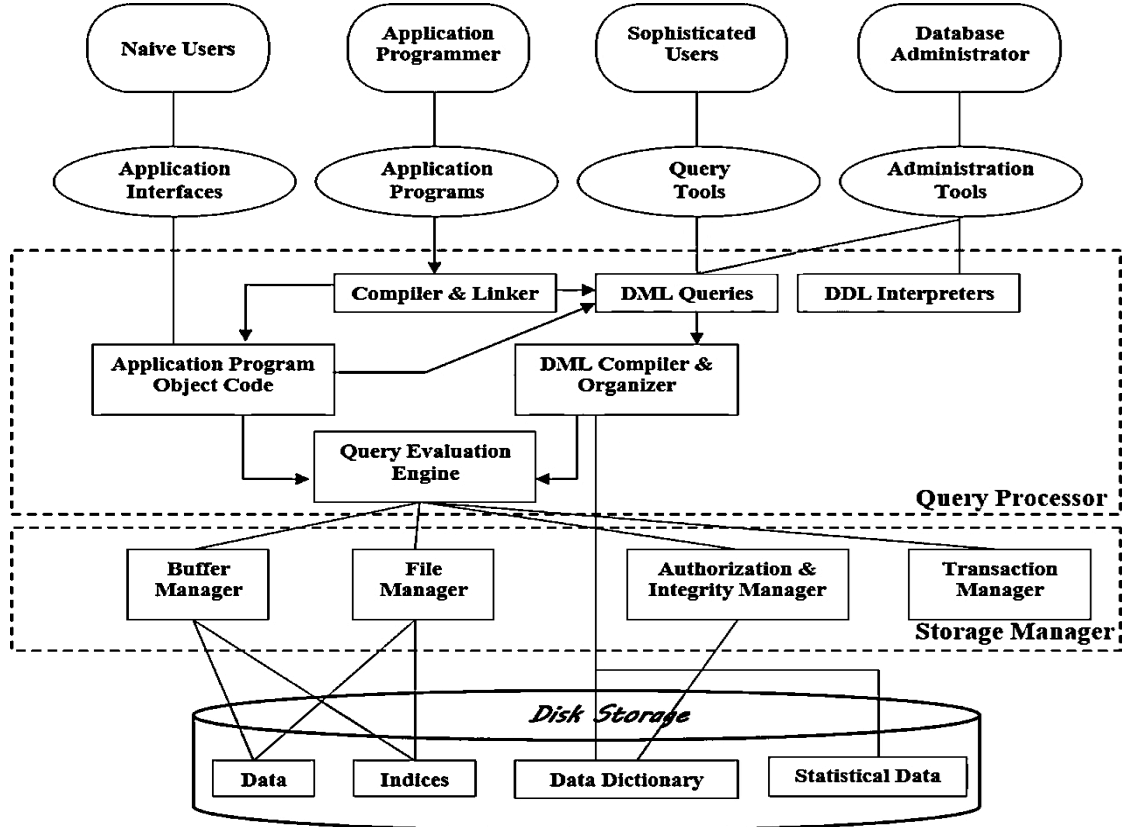
नियम 9: लॉजिकल डाटा इंडिपेंडन्स नियम (Logical Data Independence): व्ह्यू स्कीमा व्याख्येवर परिणाम न करता लॉजिकल स्कीमा व्याख्या सुधारण्याची क्षमता याला "लॉजिकल डाटा इंडिपेंडन्स" म्हणतात.

नियम 10: इंटिग्रिटी इंडिपेंडन्स नियम (Integrity Independence): अखंडतेचे निर्बंध (Integrity Constraints) ॲप्लिकेशन प्रोग्राम्समधून वेगळे नमूद केले जावे आणि कॅटलॉगमध्ये संग्रहित केले जावे. ते डाटाबेस सिस्टिमद्वारे स्वयंचलितपणे लागू केले जावे.

नियम 11: डिस्ट्रीब्यूशन इंडिपेंडन्स नियम (Distribution Independence): एकाधिक स्थानांवर डाटाचे वितरण (distribution of data across multiple locations) वापरकर्त्यासाठी अदृश्य असले पाहिजे आणि डाटाबेस सिस्टमने वितरण पारदर्शकपणे हाताळले पाहिजे.

नियम 12: नॉन-सबव्हर्जन नियम (Non-Subversion Rule): सिस्टमचा इंटरफेस सिस्टमला हानी पोहोचवू शकत नाही आणि सुरक्षा आणि अखंडतेच्या मर्यादांना बायपास करू शकत नाही.

1.1.12 डीबीएमसची एकूण रचना (Overall Structure of DBMS):



आकृती 1. 6 डीबीएमसची एकूण रचना (Overall Structure of DBMS)

1. केरी प्रोसेसर (Query Processor): हे ॲप्लिकेशन प्रोग्रामद्वारे अंतिम वापरकर्त्याकडून प्राप्त झालेल्या विनंत्या (केरी) सूचनांमध्ये स्पष्ट करते. हे डीएमएल कंपाइलरकडून प्राप्त झालेल्या वापरकर्त्याची विनंती देखील कार्यान्वित करते. केरी प्रोसेसरमध्ये खालील घटक आहेत -

- **डीएमएल कंपाइलर(DML Compiler):** हे डीएमएल स्टेटमेंट्सवर लो लेव्हल इंस्ट्रक्शन (मशीन लॅंग्वेज) मध्ये प्रक्रिया करते, जेणेकरून ते कार्यान्वित करता येतील.
- **डीडीएल इंटरप्रेटर(DDL Interpreter) :** हे मेटा डाटा (डाटा बदल डाटा) असलेल्या टेबलच्या संचामध्ये डीडीएल विधानांवर प्रक्रिया करते.
- **केरी इव्हलुएशन इंजिन (Query Evaluation Engine):** हे डीएमएल कंपाइलरद्वारे व्युत्पन्न केलेल्या सूचना कार्यान्वित करते.

2. स्टोरेज मॅनेजर(Storage Manager): स्टोरेज मॅनेजर हा एक प्रोग्राम आहे जो डाटाबेसमध्ये संग्रहित केलेला डाटा आणि प्राप्त झालेल्या केरी दरम्यान इंटरफेस प्रदान करतो. त्याला डाटाबेस कंट्रोल सिस्टम असेही म्हणतात. हे बंधने लागू करून आणि डीसीएल(DCL) स्टेटमेंट्सची अंमलबजावणी करून डाटाबेसची सातत्य आणि अखंडता राखते. डाटाबेसमधील डाटा अद्यतनित करणे, संचयित करणे, हटविणे आणि पुनर्प्राप्त करणे यासाठी ते जबाबदार आहे. त्यात खालील घटक असतात.

- **बफर मॅनेजर (Buffer Manager):** कॅशे मेमरी (Cache memory) आणि दुय्यम स्टोरेज (Secondary storage) आणि मुख्य मेमरी (Main memory) दरम्यान डाटाचे हस्तांतरण यासाठी ते जबाबदार आहे.
- **फाइल मॅनेजर (File Manager):** ते फाइल स्पेस आणि डाटाबेसमधील माहितीचे प्रतिनिधित्व करण्यासाठी वापरल्या जाणाऱ्या डाटा स्ट्रक्चरचे व्यवस्थापन करते.
- **ऑथोरायझेशन आणि इंटिग्रिटी मॅनेजर (Authorization and Integrity Manager):** हे भूमिका-आधारित प्रवेश नियंत्रण सुनिश्चित करते, म्हणजे, विनंती केलेले ऑपरेशन करण्यासाठी विशिष्ट व्यक्तीला विशेषाधिकार आहे की नाही हे तपासते. **इंटिग्रिटी मॅनेजर** जेव्हा डाटाबेस सुधारित केला जातो तेव्हा ते अखंडतेच्या मर्यादा तपासते.
- **ट्रान्झॅक्शन मॅनेजर (Transaction Manager):** तो व्यवहार प्राप्त करण्याच्या अनुसूचित पद्धतीने ऑपरेशन्स करून समवर्ती प्रवेश नियंत्रित करतो. अशा प्रकारे, हे सुनिश्चित करते की व्यवहाराच्या अंमलबजावणीपूर्वी आणि नंतर डाटाबेस सुसंगत स्थितीत राहते.

3. डिस्क स्टोरेज (Disk Storage): यात खालील घटक आहेत :

- **डाटा (Data) :** हे डाटा संग्रहित करते.
- **डाटा डिक्शनरी (Data dictionary):** यामध्ये कोणत्याही डाटाबेस ऑब्जेक्टच्या संरचनेची माहिती असते. हे माहितीचे भांडार आहे जे मेटाडाटा नियंत्रित करते.
- **निर्देशांक (Indices):** हे डाटा आयटमची जलद पुनर्प्राप्ती प्रदान करते.

डाटाबेस मॅनेजमेंट सिस्टम (DBMS) ची रचना तीन मुख्य घटकांमध्ये विभागली जाऊ शकते: अंतर्गत स्तर, संकल्पनात्मक स्तर आणि बाह्य स्तर.

अंतर्गत स्तर (Internal Level): ही पातळी डाटाबेसमधील डाटाचे भौतिक संचयन दर्शवते. हार्ड ड्राइव्ह किंवा सॉलिड-स्टेट ड्राइव्हस् यांसारख्या स्टोरेज डिव्हाइसेसमधून डाटा संग्रहित आणि पुनर्प्राप्त करण्यासाठी ते जबाबदार आहे. हे डाटा कॉम्प्रेसन, इंडेक्सिंग आणि स्टोरेज वाटप यासारख्या निम्न-स्तरीय अंमलबजावणी तपशीलांशी संबंधित आहे.

संकल्पनात्मक स्तर (Conceptual Level): ही पातळी डाटाबेसचे तार्किक दृश्य दर्शवते. हे डाटाबेसमधील डाटाच्या एकूण संस्थेशी आणि त्यांच्यातील संबंधांशी संबंधित आहे. ते डाटा स्कीमा परिभाषित करते, ज्यामध्ये सारण्या, विशेषता आणि त्यांचे संबंध समाविष्ट असतात. संकल्पनात्मक पातळी कोणत्याही विशिष्ट डीबीएमएस (DBMS) पेक्षा स्वतंत्र आहे आणि भिन्न डीबीएमएस (DBMS) वापरून अंमलात आणली जाऊ शकते.

बाह्य स्तर (External Level): ही पातळी वापरकर्त्याचे डाटाबेसचे दृश्य दर्शवते. हे वापरकर्ते डाटाबेसमधील डाटा कसा ऍक्सेस करतात याच्याशी संबंधित आहे. हे वापरकर्त्यांना अंतर्निहित अंमलबजावणी तपशीलांची चिंता न करता त्यांना अर्थपूर्ण पद्धतीने डाटा पाहण्याची परवानगी देते. बाह्य स्तर डाटाबेसला दृश्ये किंवा इंटरफेसचा संच प्रदान करते, जे विशिष्ट वापरकर्ता गटांच्या गरजा पूर्ण करण्यासाठी तयार केले जातात.

तीन स्तर एका स्कीमा मॅपिंग प्रक्रियेद्वारे जोडलेले आहेत जे डाटा एका स्तरावरून दुसऱ्या स्तरावर अनुवादित करते. स्कीमा मॅपिंग प्रक्रिया हे सुनिश्चित करते की एका स्तरावर केलेले बदल इतर स्तरांवर प्रतिबिंबित होतात. या तीन स्तरांव्यतिरिक्त, डीबीएमएस (DBMS) मध्ये डाटाबेस ॲडमिनिस्ट्रेटर (DBA) घटक देखील समाविष्ट असतो, जो डाटाबेस सिस्टम व्यवस्थापित करण्यासाठी जबाबदार असतो. डाटाबेस डिझाइन, सुरक्षा व्यवस्थापन, बॅकअप आणि पुनर्प्राप्ती आणि कार्यप्रदर्शन ट्यूनिंग यासारख्या कार्यांसाठी डीबीए (DBA) जबाबदार आहे.

एकंदरीत, डीबीएमएसची (DBMS) रचना वापरकर्त्यांना उच्च स्तरीय अमूर्तता प्रदान करण्यासाठी डिझाइन केलेली आहे, तरीही निम्न-स्तरीय अंमलबजावणी तपशील प्रभावीपणे व्यवस्थापित करण्यास अनुमती देते. हे वापरकर्त्यांना भौतिक संचयन किंवा अंमलबजावणी तपशीलांची चिंता न करता डाटाबेसमधील डाटाच्या तार्किक संस्थेवर लक्ष केंद्रित करण्यास अनुमती देते.

1.2 डीबीएमएस आर्किटेक्चर (DBMS Architecture):

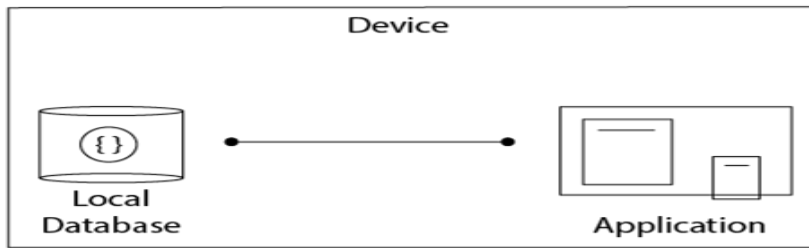
डाटा त्वरीत आणि सुरक्षितपणे ऍक्सेस करण्यासाठी डाटाबेसमध्ये बरीच महत्त्वपूर्ण माहिती साठवली जाते. म्हणून कार्यक्षम डाटा व्यवस्थापनासाठी योग्य आर्किटेक्चर निवडणे महत्वाचे आहे. डीबीएमएस(DBMS) आर्किटेक्चर वापरकर्त्यांना डाटाबेसशी कनेक्ट करताना त्यांच्या विनंत्या पूर्ण करण्यात मदत करते. डाटाबेसचा आकार, वापरकर्त्यांची संख्या आणि वापरकर्त्यांमधील संबंध यासारख्या अनेक घटकांवर अवलंबून आम्ही डाटाबेस आर्किटेक्चर निवडतो.

डीबीएमएस आर्किटेक्चरचे प्रकार(Types of DBMS Architecture):

डीबीएमएसमध्ये, आर्किटेक्चरचे तीन प्रकार आहेत, जे खालीलप्रमाणे आहेत:

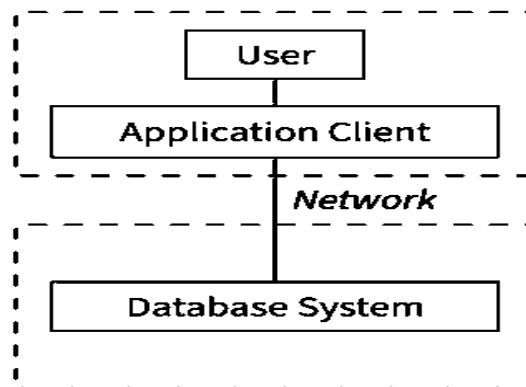
- 1- टियर आर्किटेक्चर (1-Tier Architecture)
- 2- टियर आर्किटेक्चर (2-Tier Architecture)
- 3- टियर आर्किटेक्चर (3-Tier Architecture)

1-टियर आर्किटेक्चर (1-Tier Architecture): 1-टियर आर्किटेक्चरमध्ये डाटाबेस थेट वापरकर्त्यासाठी उपलब्ध असतो, वापरकर्ता थेट डीबीएमएस (DBMS) वर बसून त्याचा वापर करू शकतो म्हणजेच क्लायंट, सर्व्हर आणि डाटाबेस सर्व एकाच मशीनवर उपस्थित असतात. उदाहरणार्थ: एसक्यूएल (SQL) शिकण्यासाठी आम्ही स्थानिक प्रणालीवर एसक्यूएल (SQL) सर्व्हर आणि डाटाबेस सेट करतो. हे आम्हाला रिलेशनल डाटाबेसशी थेट संवाद साधण्यास आणि ऑपरेशन्स कार्यान्वित करण्यास सक्षम करते. इंडस्ट्री हे आर्किटेक्चर वापरणार नाही जे ते तर्कशुद्धपणे 2-स्तरीय आणि 3-स्तरीय आर्किटेक्चरसाठी जातात.



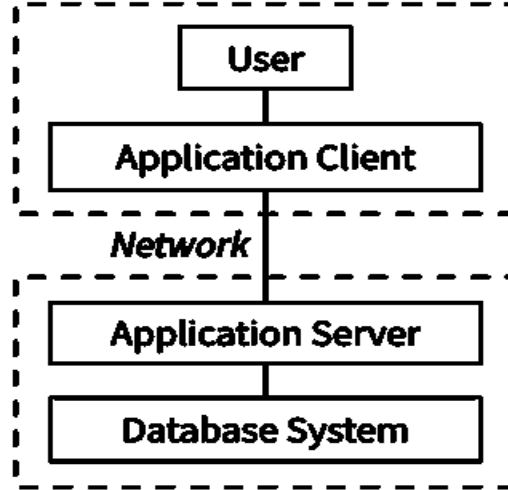
आकृती 1.7: 1 टियर आर्किटेक्चर (1-Tier Architecture)

2-टियर आर्किटेक्चर (2-Tier Architecture): 2-स्तरीय आर्किटेक्चर मूलभूत क्लायंट-सर्व्हर मॉडेलसारखेच आहे. क्लायंटच्या शेवटी असलेला अनुप्रयोग थेट सर्व्हरच्या बाजूच्या डाटाबेसशी संवाद साधतो. या परस्परसंवादासाठी ओडीबीसी(ODBC) आणि जेडीबीसी (JDBC) सारखे एपीआय (API) वापरले जातात. सर्व्हर साइड केरी प्रक्रिया आणि व्यवहार व्यवस्थापन कार्ये प्रदान करण्यासाठी जबाबदार आहे. क्लायंटच्या बाजूने, वापरकर्ता इंटरफेस आणि अनुप्रयोग प्रोग्राम चालवले जातात. क्लायंटच्या बाजूचा अनुप्रयोग डीबीएमएसशी (DBMS) संवाद साधण्यासाठी सर्व्हरच्या बाजूने कनेक्शन स्थापित करतो.



आकृती 1. 8: 2 टियर आर्किटेक्चर (2-Tier Architecture)

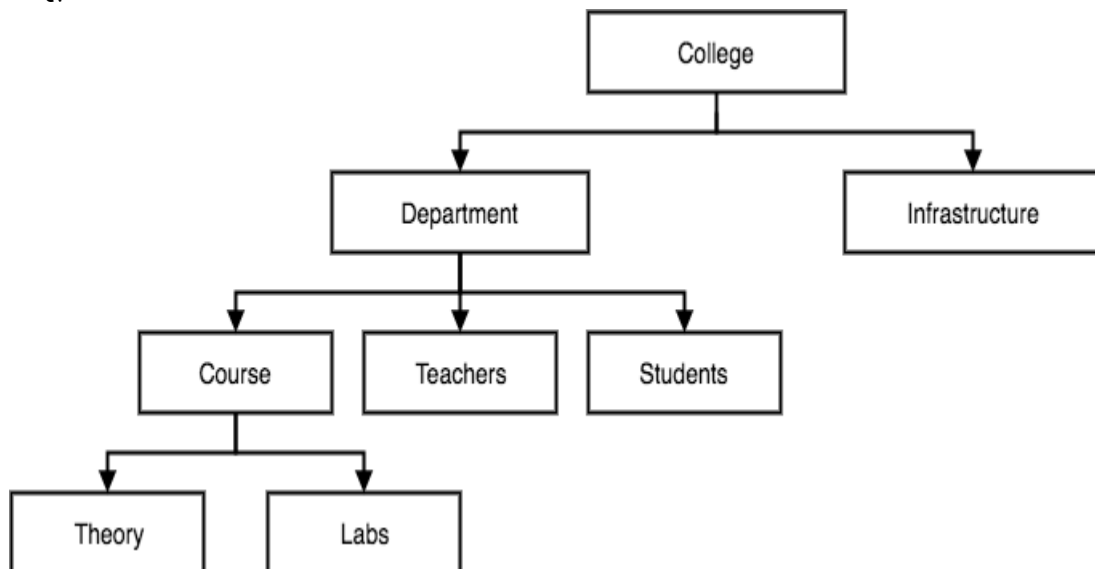
3-टियर आर्किटेक्चर (3-Tier Architecture): 3-टियर आर्किटेक्चरमध्ये, क्लायंट आणि सर्व्हरमध्ये आणखी एक स्तर असतो. क्लायंट थेट सर्व्हरशी संवाद साधत नाही. त्याऐवजी, ते ॲप्लिकेशन सर्व्हरशी संवाद साधते जे पुढे डाटाबेस सिस्टमशी संवाद साधते आणि नंतर क्लेरी प्रक्रिया आणि व्यवहार व्यवस्थापन होते. हा मध्यवर्ती स्तर सर्व्हर आणि क्लायंट दरम्यान अंशतः प्रक्रिया केलेल्या डाटाच्या देवाणघेवाणीसाठी एक माध्यम म्हणून कार्य करतो. मोठ्या वेब ॲप्लिकेशन्सच्या बाबतीत या प्रकारच्या आर्किटेक्चरचा वापर केला जातो.



आकृती 1. 9 3 टियर आर्किटेक्चर (3-Tier Architecture)

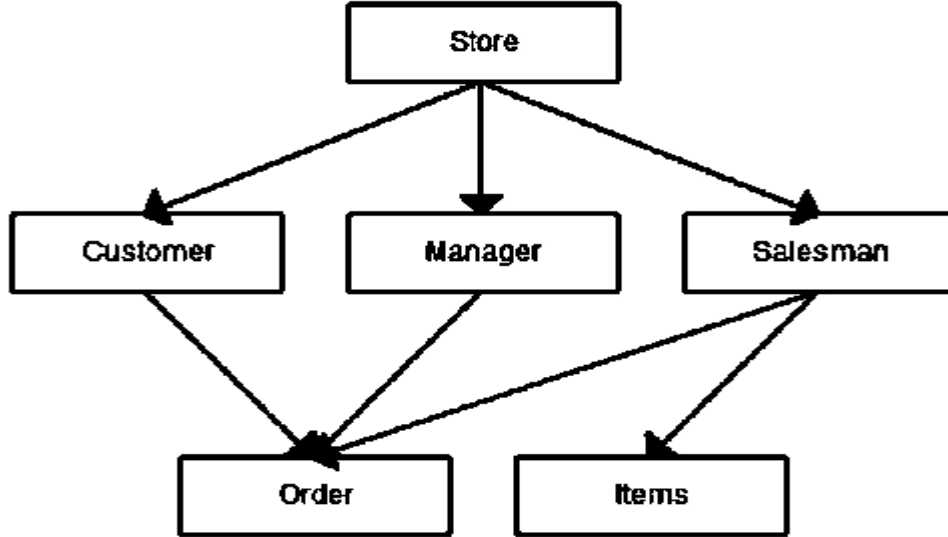
1.3 डाटा मॉडेल (Data Models): डाटाबेस मॅनेजमेंट सिस्टम (DBMS) मधील डाटा मॉडेल ही डाटाबेसच्या वर्णनाचा सारांश देण्यासाठी विकसित केलेल्या साधनांची संकल्पना आहे. डाटा मॉडेल्स आम्हाला डाटाचे पारदर्शक चित्र प्रदान करतात जे आम्हाला वास्तविक डाटाबेस तयार करण्यात मदत करतात.

1.3.1. हिरअर्चिकल मॉडेल (Hierarchical Model) : हे डाटा मॉडेलमधील सर्वात जुने मॉडेल आहे जे 1950 मध्ये IBM ने विकसित केले होते. श्रेणीबद्ध मॉडेलमध्ये, डाटा सारण्यांचा संग्रह म्हणून पाहिला जातो किंवा आपण श्रेणीबद्ध संबंध तयार करणारे विभाग म्हणू शकतो. यामध्ये, डाटा एका झाडासारख्या संरचनेत आयोजित केला जातो जेथे प्रत्येक रेकॉर्डमध्ये एक पालक रेकॉर्ड आणि अनेक मुले असतात. जरी सेगमेंट लॉजिकल असोसिएशनद्वारे साखळीसारखी रचना म्हणून जोडलेले असले तरीही, झटपट रचना अनेक शाखांसह फॅन स्ट्रक्चर असू शकते. अतार्किक संघटनांना आपण दिशात्मक संघटना म्हणतो.



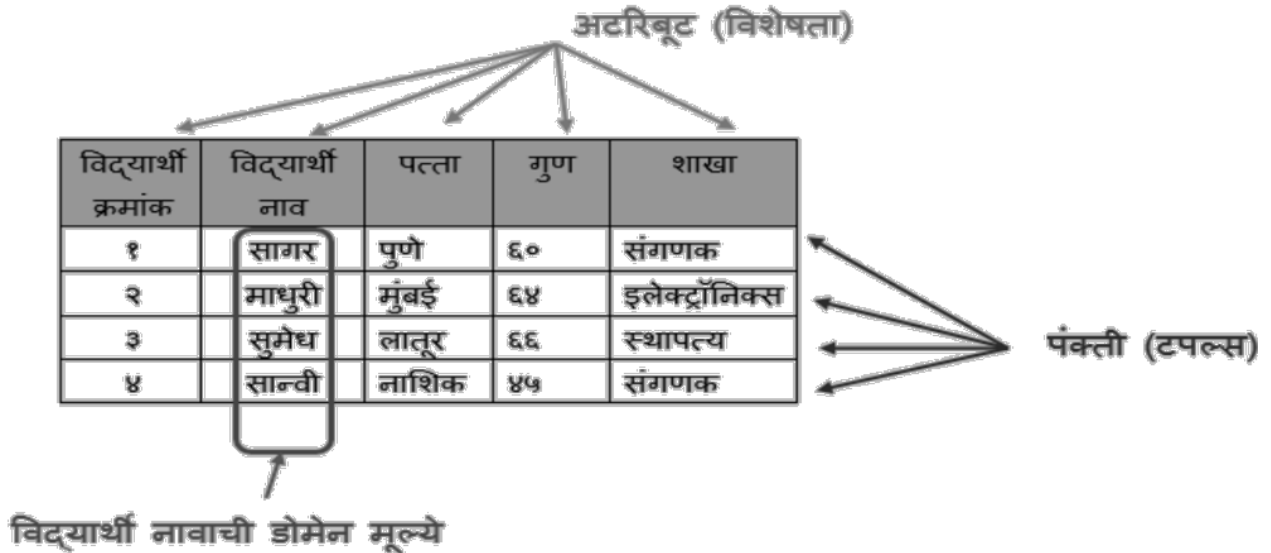
आकृती 1.10: हिरअर्चिकल मॉडेल (Hierarchical Model)

2. नेटवर्क मॉडेल (Network Model): नेटवर्क मॉडेल 1960 च्या दशकात डाटाबेस टास्क ग्रुपने औपचारिक केले होते. हे मॉडेल श्रेणीबद्ध मॉडेलचे सामान्यीकरण आहे. या मॉडेलमध्ये एकाधिक पालक विभाग असू शकतात आणि हे विभाग स्तर म्हणून गटबद्ध केले आहेत परंतु कोणत्याही स्तराशी संबंधित विभागांमध्ये तार्किक संबंध आहे. मुख्यतः, कोणत्याही दोन विभागांमध्ये अनेक-ते-अनेक तार्किक संबंध अस्तित्वात असतात.



आकृती 1.11: नेटवर्क मॉडेल (Network Model)

3. रिलेशनल मॉडेल (Relational Model) : संबंधात्मक डाटाबेस हा एक प्रकारचा डाटाबेस आहे, ज्यात पंक्ती (रो (Row)) आणि स्तंभ (कॉलम (coloum)) वापरून रचनात्मक / स्ट्रक्चर्ड (structured) स्वरूपात माहिती संग्रहित होते. यामुळे डाटाबेसमध्ये विशिष्ट माहितीचा शोध घेणे आणि मिळविणे सोपे होते. यामध्ये प्रत्येक सारणीमधील (टेबलमधील) माहिती एकमेकांशी संबंधित असते आणि इतर सारण्यांशी देखील संबंधित असू शकते.



आकृती 1.12: रिलेशनल मॉडेल (Relational Model)

References:

1. <https://nptel.ac.in/courses/106105175>
2. <https://www.w3schools.com/sql/>
3. <https://www.tutorialspoint.com/sql/index.htm>
4. Database System Concepts by Henry F. Korth McGraw Hill Education

युनिट - 2

रिलेशनल डाटा मॉडेल

(Relational Data Model)

विषय निष्पत्ती (Course Outcome): CO2 दिलेल्या समस्येसाठी डाटाबेस डिझाइन करा.

घटक निष्पत्ती (Theory Learning Outcome):

1. डाटाबेसची रिलेशनल स्ट्रक्चर स्पष्ट करा.
2. उदाहरणांसह कीचे प्रकार लिहा.
3. दिलेल्या समस्येसाठी ER आकृती काढा.
4. विविध सामान्यीकरण रूपे (नॉर्मलिझेशन फॉर्म्स) स्पष्ट करा.

2.1 रिलेशनल रचना (Relational Structure):

1. टेबल (रिलेशन) (Table/Relation): टेबल एक डाटाबेस ऑब्जेक्ट (Database Object) आहे, जो पंक्ती (Row) आणि स्तंभांच्या (Column) स्वरूपात दिलेल्या गोष्टी (डाटा) संग्रहित करतो.

उदाहरणार्थ: विद्यार्थी टेबल

स्तंभ (Column)				
विद्यार्थी क्रमांक	विद्यार्थी नाव	पत्ता	गुण	शाखा
१	सागर	पुणे	६०	संगणक
२	माधुरी	मुंबई	६४	इलेक्ट्रॉनिक्स
३	सुमेध	लातूर	६६	स्थापत्य
४	सान्वी	नाशिक	४५	संगणक

पंक्ती (Row)

आकृती 2.1 टेबल (Table)

2. पंक्ती (टपल्स) (Row / Tuple): टेबलच्या प्रत्येक पंक्तीला 'डाटा रेकॉर्ड' म्हणतात. टपल, ज्याला 'रेकॉर्ड' किंवा 'रो' म्हणूनही ओळखले जाते, हे रिलेशनल डाटाबेस मॅनेजमेंट सिस्टम (DBMS) मधील डाटाचे मूलभूत एकक (basic unit) आहे. उदाहरणार्थ: विद्यार्थी टेबलमध्ये खालील पंक्ती आहेत.

१	सागर	पुणे	६०	संगणक
२	माधुरी	मुंबई	६४	इलेक्ट्रॉनिक्स
३	सुमेध	लातूर	६६	स्थापत्य
४	सान्वी	नाशिक	४५	संगणक

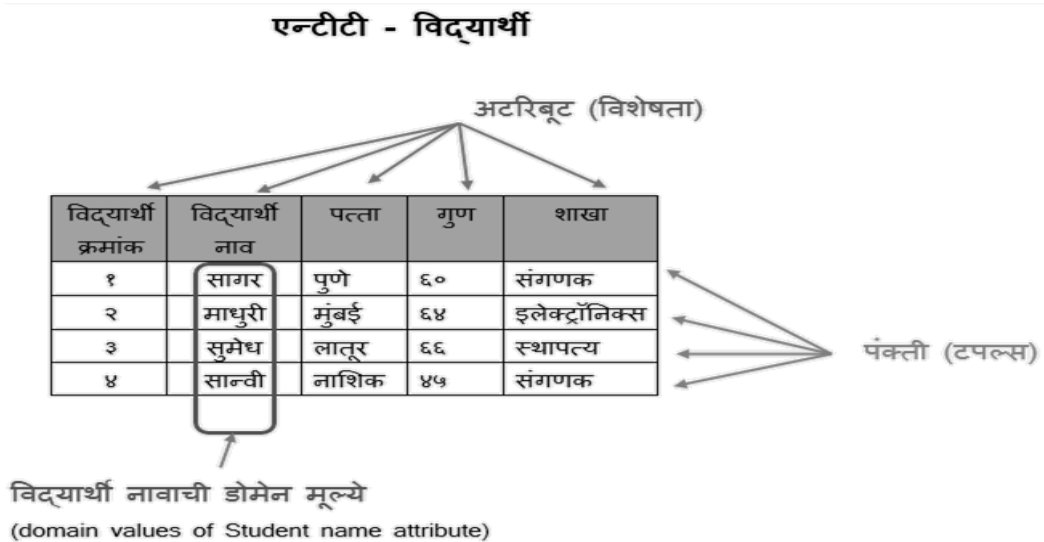
3. डोमेन (Domain): डाटाबेस व्यवस्थापनामध्ये, डोमेन हा मूल्यांचा एक संच असतो जो डाटाबेसच्या स्तंभामध्ये संग्रहित केला जाऊ शकतो. डाटाबेस डोमेनची साधी व्याख्या म्हणजे डाटाबेसमधील स्तंभाद्वारे वापरलेला डाटा प्रकार.

उदाहरणार्थ: विद्यार्थी टेबलमध्ये विद्यार्थी क्रमांक डोमेन 1,2, 3, 4 ही अंकीय मूल्ये संग्रहित करते.

4. अट्रीब्युट (गुणधर्म / विशेषता) (Attribute): अट्रीब्युट(विशेषता) डाटाबेस टेबलमधील स्तंभ किंवा फील्डचा संदर्भ देते. अट्रीब्युट(विशेषता) हा डाटाचा एक भाग आहे जो एखाद्या घटकाचे (Entity) वर्णन करतो. उदाहरणार्थ: विद्यार्थी टेबलमध्ये -विद्यार्थी क्रमांक, विद्यार्थी नाव, पत्ता, गुण आणि शाखा हे अट्रीब्युट(विशेषता) आहेत.

5. एन्टीटी (Entity): एन्टीटी, एखादी भौतिक अस्तित्व असलेली वस्तू असू शकते जसे कि - एक विशिष्ट व्यक्ती, कार, घर, कर्मचारी इत्यादी. किंवा एखादी एन्टीटी संकल्पनात्मक अस्तित्व असलेली एखादी वस्तू असू शकते जसे कि - एक कंपनी, नोकरी, विद्यापीठाचा अभ्यासक्रम इत्यादी.

एन्टीटी ही अशी गोष्ट आहे जी इतर गोष्टींपासून वेगळी असते. तिची स्वतःची स्पष्ट ओळख असते. एन्टीटी ही एक गोष्ट, व्यक्ती, ठिकाण किंवा वस्तू असू शकते. एन्टीटी ला एक किंवा जास्त अट्रिब्यूट (विशेषता) असू शकतात. उदाहरणार्थ: विद्यार्थी ही एक एन्टीटी असून -विद्यार्थी क्रमांक, विद्यार्थी नाव, पत्ता, गुण आणि शाखा हे तिचे अट्रिब्यूट (विशेषता) आहेत.



आकृती 2.2 एन्टीटी (Entity)

2.2 किज (Keys)

1. सुपर की (Super Key): विशिष्टपणे रेकॉर्ड ओळखण्यासाठी वापरल्या जाणाऱ्या अट्रीब्युट(विशेषता) किंवा अट्रीब्युटच्या संयोजनाला सुपर की म्हणतात.

उदाहरणार्थ:

दिलेल्या विद्यार्थी टेबलसाठी सुपर की खालीलप्रमाणे,

सुपर की 1: { विद्यार्थी क्रमांक }

सुपर की 2: { विद्यार्थी क्रमांक, विद्यार्थी नाव }

सुपर की 3: { विद्यार्थी क्रमांक, विद्यार्थी नाव, गुण }

2. कॅंडिडेट की (Candidate Key): काहीवेळा, एकापेक्षा जास्त अट्रीब्युटमध्ये अद्वितीय ओळख गुणधर्म असू शकतात. या अट्रीब्युटना कॅंडिडेट की म्हणून ओळखले जाते.

उदाहरणार्थ:

विद्यार्थी टेबलसाठी, जर विद्यार्थी क्रमांक आणि विद्यार्थी नाव दोन्ही अद्वितीय (unique) असतील तर त्या दोघांना कॅंडीडेट की म्हणून ओळखले जाते.

कॅंडीडेट की 1: { विद्यार्थी क्रमांक }

कॅंडीडेट की 2: { विद्यार्थी नाव }

3. प्रायमरी की (Primary Key): रिलेशनल टेबलमध्ये, नेहमी एक अट्रीब्युट असते ज्यात अद्वितीय ओळख (यूनिक) मूल्ये क्षमता असते. ते टेबल मधील प्रत्येक पंक्ती (टपल) युनिकली ओळखू शकते. अशा यूनिक अट्रीब्युटला प्रायमरी की म्हणतात.

उदाहरणार्थ:

विद्यार्थी टेबलसाठी,

प्रायमरी की: { विद्यार्थी क्रमांक }

4. फॉरेन की (Foreign Key): डीबीएमएस (डाटाबेस मॅनेजमेंट सिस्टीम) मधील फॉरेन की हे एक फील्ड आहे जे दोन वेगळ्या टेबलमधील दुवा स्थापित करते आणि राखते. फॉरेन की ही एका टेबलमधील प्रायमरी की असते आणि ती दुसऱ्या टेबलमध्ये फील्ड म्हणून दिसते. फॉरेन की एका टेबलमधील डाटाला दुसऱ्या टेबलमधील डाटाशी जोडते.

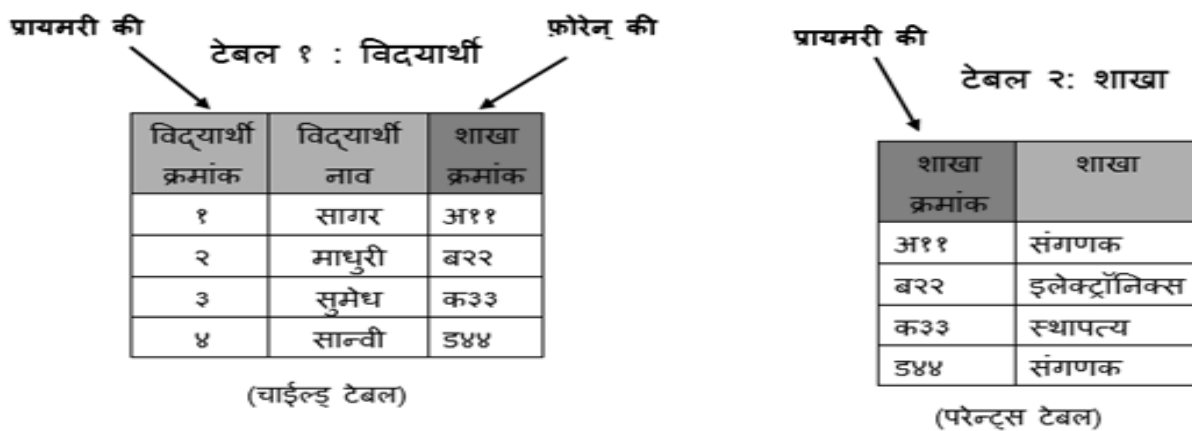
उदाहरणार्थ:

विद्यार्थी टेबलसाठी,

प्रायमरी की: { विद्यार्थी क्रमांक }

फॉरेन की: { शाखा क्रमांक }

शाखा क्रमांक ही शाखा टेबलमधील प्रायमरी की आहे आणि ती विद्यार्थी टेबलमध्ये फील्ड म्हणून दिसते.



आकृती 2.3 प्रायमरी की (Primary Key) आणि फॉरेन की (Foreign Key)

2.3 डाटा कंस्ट्रेंट (Data Constraints): टेबलमधील डाटासाठी नियम निर्दिष्ट करण्यासाठी डाटा कंस्ट्रेंटचा (डाटा मर्यादांचा) वापर केला जातो. डाटा कंस्ट्रेंट पुढील प्रकारचे आहेत :

1. डोमेन कंस्ट्रेंट (Domain Constraint)
2. रेफरेंशीयल इंटेग्रिटी कंस्ट्रेंट (Referential Integrity Constraint)

2.3.1. डोमेन कंस्ट्रेंट (Domain Constraint): डोमेन कंस्ट्रेंट, हे वापरकर्त्याच्या निर्देशानुसार (as per user specification) मूल्य राखण्यासाठी वापरले जाते.

डोमेन कंस्ट्रेंट दोन प्रकारचे आहेत:

1. नॉट नल कंस्ट्रेंट (Not Null Constraint)
2. चेक कंस्ट्रेंट (Check constraint)

1. **नॉट नल कंस्ट्रेंट (Not Null Constraint):** रिक्त मूल्य (Null Values) टाळण्यासाठी रकान्यावर नॉट नल कंस्ट्रेंट लागू केले जाते. नॉट नल कंस्ट्रेंट स्तंभात रिक्त मूल्य (Null Value) असू शकत नाही याची नेहमी खात्री करते.

उदाहरणार्थ (Example):

```
create table Student(
rollno number(3),
stud_name varchar(10) not null,
address varchar(20),
marks number(3) ,
branch varchar(20)
);
```

दिलेल्या SQL query मध्ये stud_name या अट्रीब्युटवर नॉट नल कंस्ट्रेंट वापरले आहे, म्हणून stud_name हा स्तंभ रिक्त मूल्य (null value) स्वीकारणार नाही.

2. **चेक कंस्ट्रेंट (Check Constraint):** चेक कंस्ट्रेंट स्तंभामध्ये ठेवता येणारी मूल्य श्रेणी (value range) मर्यादित (limit) करण्यासाठी वापरली जाते. चेक कंस्ट्रेंट स्तंभातील मूल्ये विशिष्ट स्थितीचे समाधान (satisfy specific condition) करतात याची खात्री करते.

उदाहरणार्थ (Example):

```
create table Student(
rollno number(3),
stud_name varchar(10),
address varchar(20),
marks number(3) check (marks>=45),
branch varchar(20)
);
```

दिलेल्या SQL query मध्ये marks या अट्रीब्युटवर चेक कंस्ट्रेंट वापरले आहे. प्रत्येक स्टुडन्ट चे marks 45 किंवा 45 पेक्षा जास्त असतील याची खात्री चेक कंस्ट्रेंट देते .

2.3.2 रेफरेंशियल इंटेग्रिटी कंस्ट्रेंट (Referential Integrity Constraint): हे दोन टेबल्समध्ये निर्दिष्ट केले जाऊ शकते. रेफरेंशियल इंटेग्रिटी कंस्ट्रेंटला फॉरेन की कंस्ट्रेंट असेही म्हणतात. फॉरेन की ही एका टेबलमधील प्रायमरी की असते आणि ती दुसऱ्या टेबलमध्ये फील्ड म्हणून दिसते. फॉरेन की एका टेबलमधील डाटाला दुसऱ्या टेबलमधील डाटाशी जोडते.

प्रायमरी की			फॉरेन की	
विद्यार्थी क्रमांक (rollno)	विद्यार्थी नाव (stud_name)	शाखा क्रमांक (course_id)	शाखा क्रमांक (course_id)	शाखा (course_name)
१	सागर	अ११	अ११	संगणक
२	माधुरी	ब२२	ब२२	इलेक्ट्रॉनिक्स
३	सुमेध	क३३	क३३	स्थापत्य
४	सान्वी	ड४४	ड४४	संगणक

टेबल १: विद्यार्थी (Student)
[चाईल्ड टेबल]

टेबल २: शाखा (Course)
[पॅरेंट्स टेबल]

आकृती 2.4 प्रायमरी की (Primary Key) आणि फॉरेन की (Foreign Key) चे उदाहरण
शाखा (Course) टेबल, हे पॅरेंट टेबल असून, ते बनवण्याची SQL query खाली नमूद केली आहे.

```
create table Course(
```

```
course_id varchar(3) primary key,
course_name varchar(10),
);
```

विद्यार्थी (Student) टेबल, हे चाईल्ड टेबल असून, ते बनवण्याची SQL query खाली नमूद केली आहे

```
create table Student(
rollno number(3) primary key,
stud_name varchar(10),
course_id varchar(3) FOREIGN KEY REFERENCES Course (course_id)
);
```

2.4 एन्टीटी रिलेशनशिप मॉडेल (Entity Relationship Model): एन्टीटी रिलेशनशिप मॉडेल हे डाटाबेसमध्ये दर्शविल्या जाणाऱ्या घटकांना (entities) ओळखण्यासाठी आणि ते कसे संबंधित (relational) आहेत याचे प्रतिनिधित्व करण्यासाठी एक मॉडेल आहे. ER मॉडेलचा वापर वास्तविक-जगातील वस्तू (जसे की व्यक्ती, कार किंवा कंपनी) आणि या वास्तविक-जगातील वस्तूंमधील संबंध तयार करण्यासाठी केला जातो. एन्टीटी रिलेशनशिप डायग्राम (ER diagram) डाटाबेसमध्ये उपस्थित असलेल्या घटकांमधील संबंध स्पष्ट करतो. थोडक्यात, ईआर डायग्राम (ER diagram) हे डाटाबेसचे संरचनात्मक स्वरूप आहे.

डीबीएमएस (DBMS) मध्ये ईआर डायग्राम (ER Diagram) का वापरावेत ?

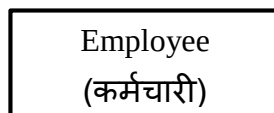
1. ईआर डायग्राम (ER diagram) मुळे टेबल रूपांतरित करणे सोपे होते.
2. हे रेखाचित्र समजण्यास अतिशय सोपे आहेत.
3. अगदी साध्या वापरकर्त्यासाठी देखील हे रेखाचित्र तयार करणे सोपे आहे.
4. हे तार्किकदृष्ट्या डाटाचे व्हिज्युअलाइझ करण्यासाठी एक मानक देते.

वरील नमूद केलेल्या कारणांमुळे डीबीएमएस (DBMS) मध्ये ईआर डायग्राम (E.R. diagram) वापरतात.

2.4.1 मजबूत एन्टीटी संच (Strong Entity Set): मजबूत एन्टीटी इतर एन्टीटी वर अवलंबून नसते. यात एक प्राथमिक की (primary key) असते, जी प्रत्येक रेकॉर्ड अद्वितीयपणे (uniquely) ओळखण्यात मदत करते.

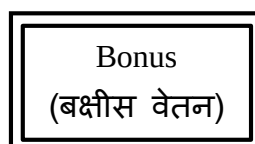
मजबूत एन्टीटी आयताद्वारे दर्शविली जाते.

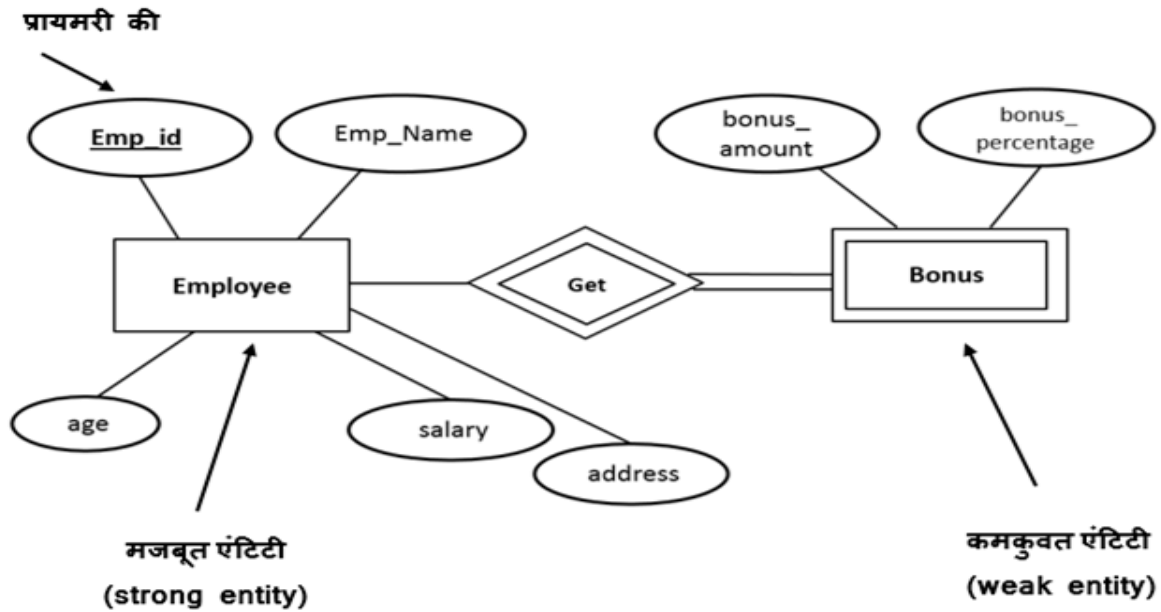
उदाहरणार्थ:



2.4.2 कमकुवत एन्टीटी संच (Weak Entity Set): कमकुवत एन्टीटी इतर एन्टीटी वर अवलंबून असते. यात प्राथमिक की (Primary Key) नसते. कमकुवत एन्टीटी दुहेरी आयताद्वारे दर्शविली जाते. कमकुवत एन्टीटी नेहमी कमकुवत रिलेशनशिप बरोबर दुहेरी चौकट आणि दुहेरी रेषा वापरून जोडली जाते.

उदाहरणार्थ:





आकृती 2.5 उदाहरण दाखल : मजबूत आणि कमकुवत एन्टीटी ईआर डायग्राम
(Example of Strong and weak entity ER diagram)

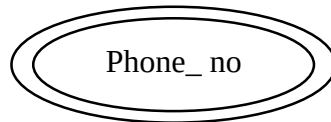
2.4.3 अट्रीब्युटचे प्रकार (Types of Attributes):

एकल-मूल्य गुणधर्म (Single-Valued Attribute): एकच मूल्य घेणारा गुणधर्म हा एकल-मूल्य गुणधर्म असतो.
उदाहरणार्थ:



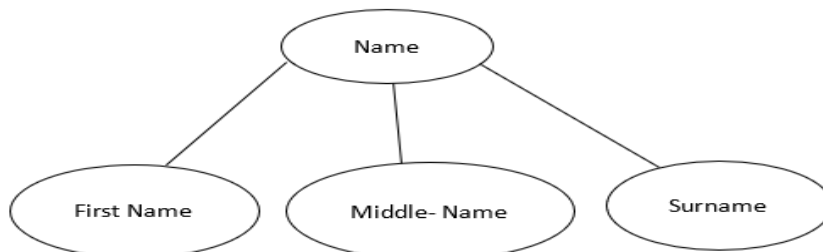
बहुमूल्य गुणधर्म (Multivalued Attribute): दिलेल्या घटकासाठी एकापेक्षा जास्त मूल्य घेणारा गुणधर्म हा बहुमूल्य गुणधर्म असतो.

उदाहरणार्थ:



संयुक्त गुणधर्म (Composite Attribute): ज्या गुणधर्मांना उपभागांमध्ये विभागले जाऊ शकते त्यांना संयुक्त गुणधर्म म्हणून ओळखले जाते.

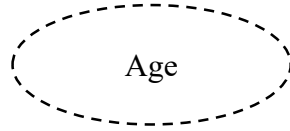
उदाहरणार्थ: Name हा संयुक्त गुणधर्म आहे.



आकृती 2.6 संयुक्त गुणधर्म (Composite attribute)

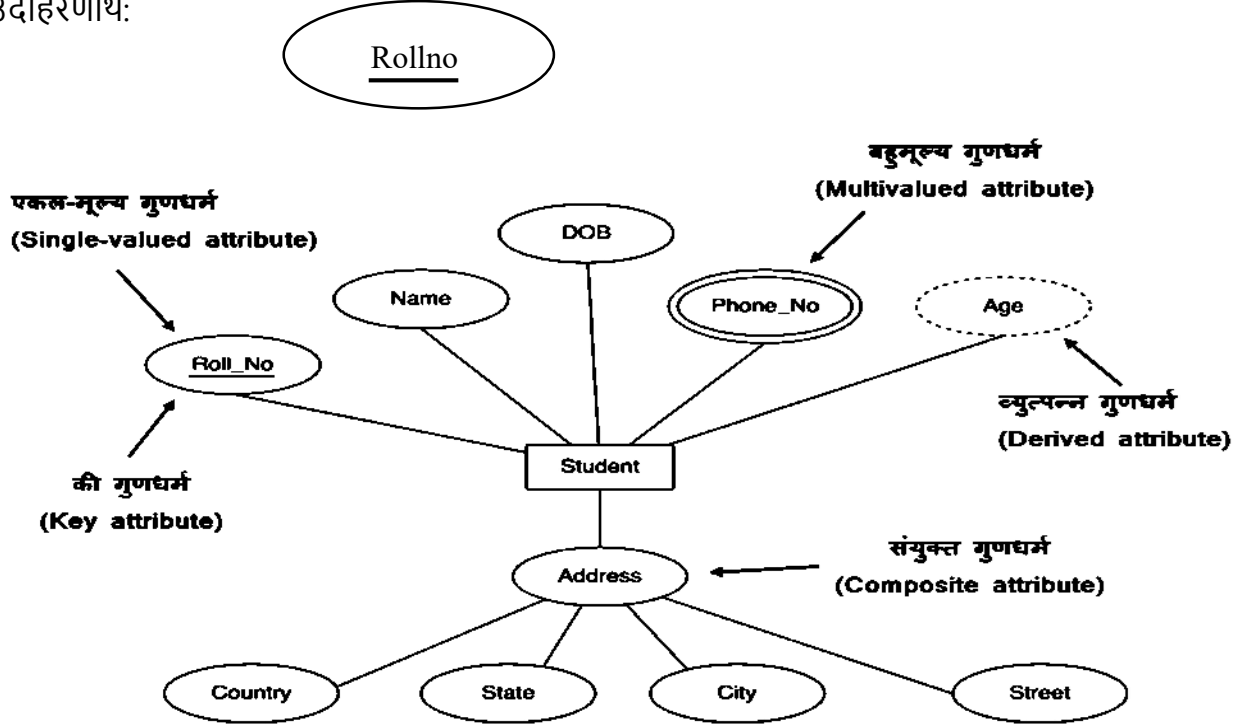
व्युत्पन्न गुणधर्म (Derived Attribute): व्युत्पन्न गुणधर्म इतर गुणधर्मांमधून मिळवता येतो.

उदाहरणार्थ: वय या गुणधर्माचे मूल्य जन्म तारखे मधून काढता येते .



मुख्य गुणधर्म/ की गुणधर्म (Key Attribute): मुख्य गुणधर्म/ की गुणधर्म ही अशी वैशिष्ट्ये आहेत जी घटक संचातील अस्तित्व अद्वितीयपणे ओळखू शकतात.

उदाहरणार्थ:



आकृती 2.7 उदाहरण दाखल: स्टुडेंट एन्टीटी आणि तिचे विविध प्रकारचे अट्रीबुट
(Example of Student entity with various attribute types)

2.4.4 ईआर डायग्राम चिन्हे (Symbols of ER Diagram)

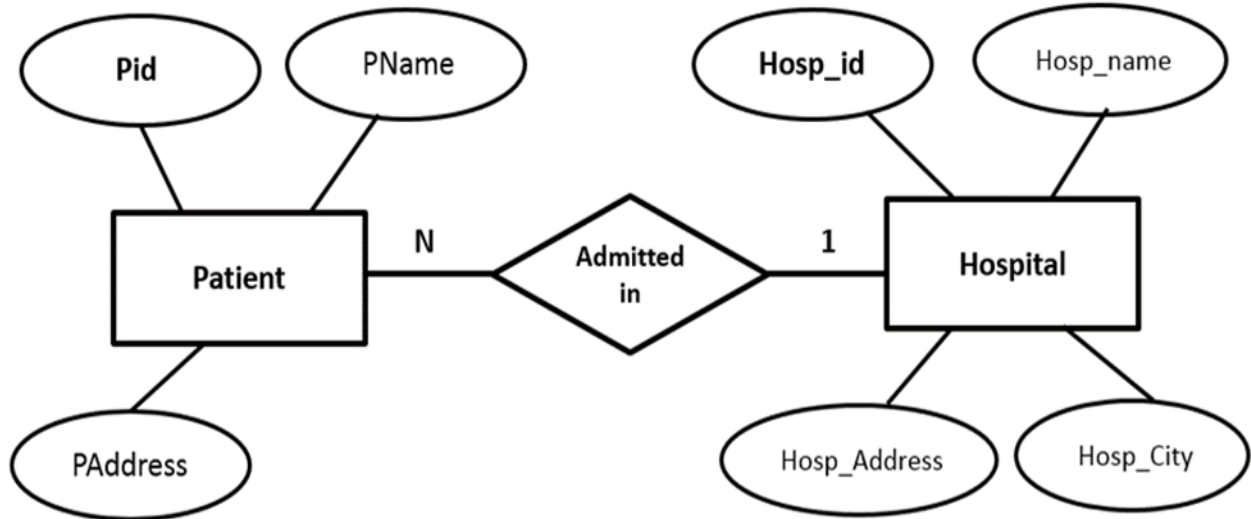
ईआर डायग्राम मध्ये खालील चिन्हे असतात:

आकृती	चिन्ह	उद्देश / वापर
आयत		आयत ER मॉडेलमधील एन्टीटी / घटक (entity) दर्शवतात.
लंबवर्तुळ		लंबवर्तुळ ER मॉडेलमधील अट्रीब्युट/गुणधर्म(attribute) दर्शवतात.
समभुज चौकोन / चौकट		चौकट ER मॉडेलमधील घटकांमधील नातेसंबंध (relationship) दर्शवतात.
रेषा		रेषा घटकांना (entities) विशेषतांना (attributes) आणि नातेसंबंध (relationship) जोडतात.
दुहेरी लंबवर्तुळ		दुहेरी लंबवर्तुळ बहु-मूल्य असलेले अट्रीब्युट(Multivalued attribute) दर्शवतात.

आकृती	चिन्ह	उद्देश / वापर
दुहेरी आयत		दुहेरी आयत कमकुवत एन्टीटी (Weak entity) दर्शवते.

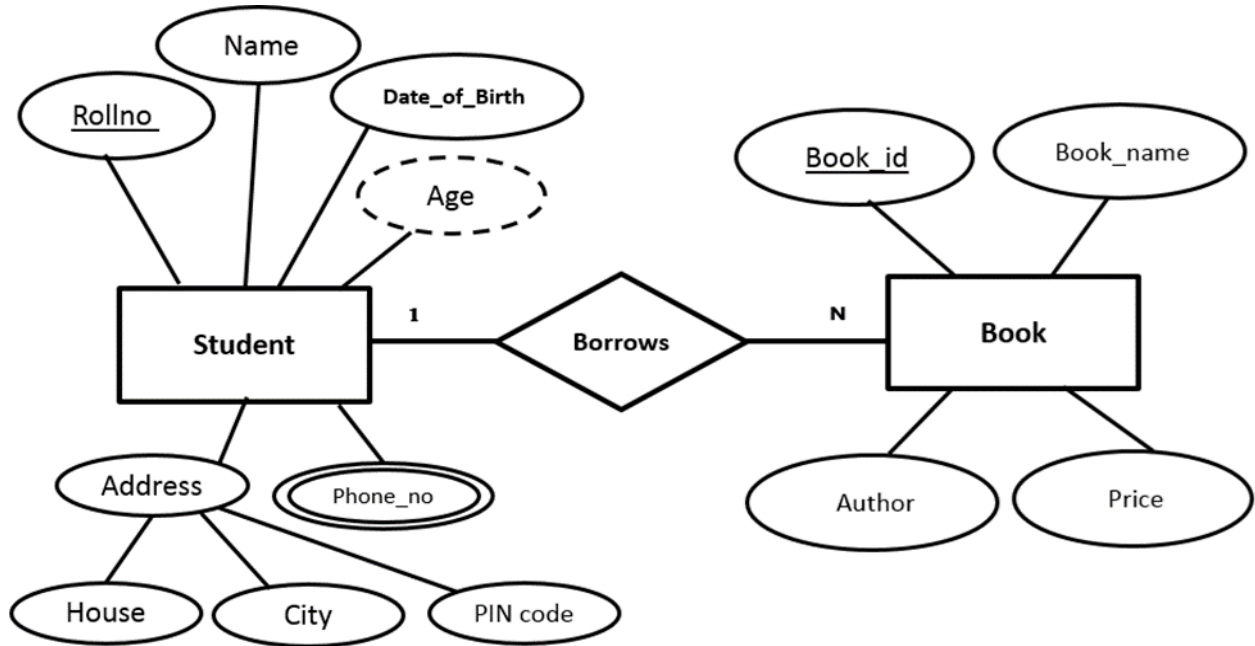
2.5.1 एन्टीटी रिलेशनशिप आकृती (Entity Relationship Diagram):

1. हॉस्पिटल मॅनेजमेंट सिस्टिम साठी E.R diagram



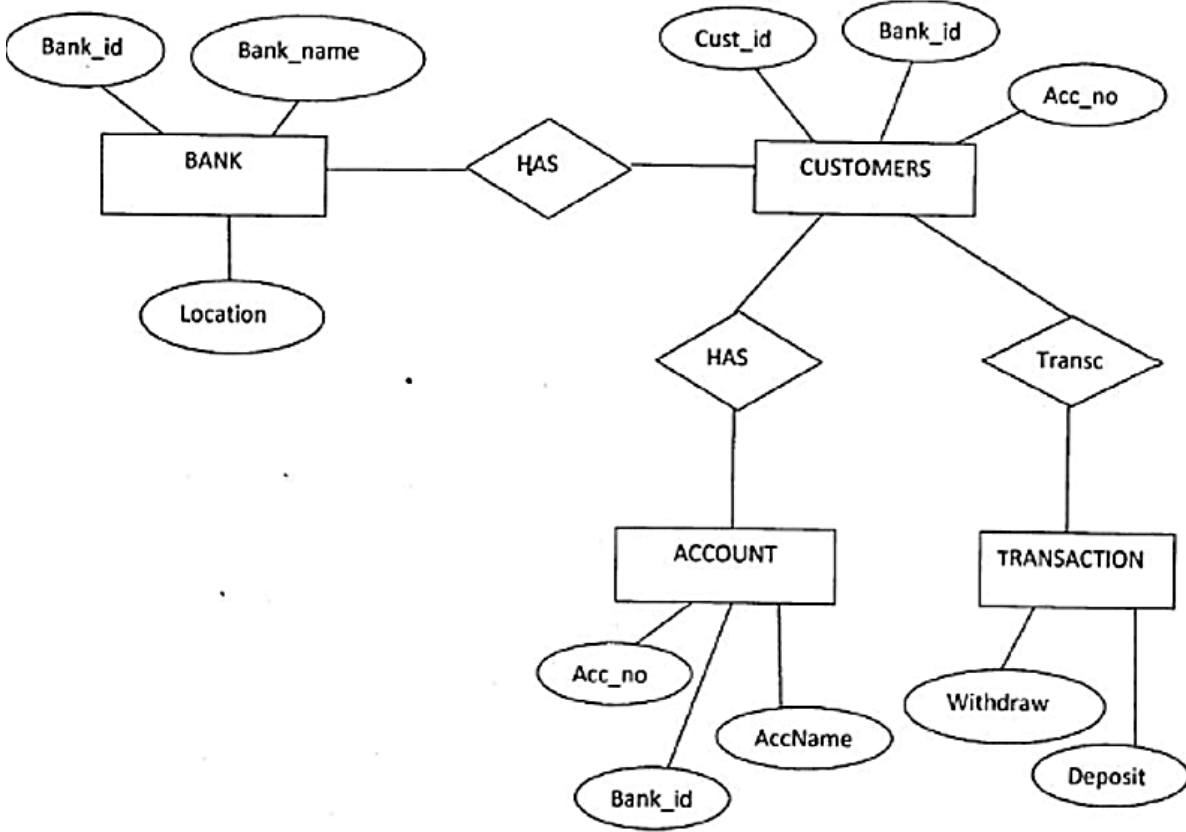
आकृती 2.8 हॉस्पिटल मॅनेजमेंट सिस्टिम साठी E.R Diagram

2. लायब्ररी मॅनेजमेंट सिस्टिम साठी E.R Diagram:



आकृती 2.9 लायब्ररी मॅनेजमेंट सिस्टिम साठी E.R Diagram

3. बँक मॅनेजमेन्ट सिस्टिम साठी E.R Diagram:



आकृती 2.10 बँक मॅनेजमेन्ट सिस्टिम साठी E.R Diagram

2. 5 सामान्यीकरण (Normalization):

सामान्यीकरण ही टेबल मधील अनावश्यकता कमी करण्याची प्रक्रिया आहे. डाटाबेस टेबलमधील रिडंडंसी (redundancy) दूर करण्यासाठी किंवा कमी करण्यासाठी सामान्य फॉर्म (Normal Forms) वापरले जातात. सामान्य फॉर्मचे प्रकार खाली सूचीबद्ध केल्याप्रमाणे आहेत –

1. पहिला सामान्य फॉर्म (1NF)
2. दुसरा सामान्य फॉर्म (2NF)
3. तिसरा सामान्य फॉर्म (3NF)

2. 5. 1 कार्यात्मक अवलंबित्व (Functional Dependency):

कार्यात्मक अवलंबित्व ही एक संकल्पना (concept) आहे जी गुणधर्मांच्या दोन संचामधील संबंध निर्दिष्ट करते. (specifies relationship between two set of attributes)

हे $X \rightarrow Y$ म्हणून दर्शविले जाते.

X ला निर्धारक (Determinant) म्हणतात, आणि Y ला अवलंबित (Dependent) म्हणतात.

कार्यात्मक अवलंबनांचे प्रकार (Types of Functional Dependencies)

1. क्षुल्लक कार्यात्मक अवलंबित्व (Trivial Functional Dependency)
2. गैर-क्षुल्लक कार्यात्मक अवलंबित्व (Non-Trivial Functional Dependency)
3. बहुमूल्य कार्यात्मक अवलंबित्व (Multivalued Functional Dependency)
4. संक्रमणात्मक कार्यात्मक अवलंबित्व अवलंबित (Transitive Functional Dependency)
5. पूर्णपणे कार्यात्मक अवलंबित्व (Fully Functional Dependency)

1. क्षुल्लक कार्यात्मक अवलंबित्व (Trivial Functional Dependency):

अवलंबित (Dependent) हा नेहमी निर्धारकाचा (Determinant) उपसंच (subset) असतो.

जर $X \rightarrow Y$ आणि Y हा X चा उपसंच असेल. तर त्याला क्षुल्लक कार्यात्मक अवलंबन म्हणतात.

उदाहरणार्थ:

विद्यार्थी (Student) टेबल

roll_no	name	age
42	abc	17
43	pqr	18
44	xyz	18

येथे, $\{roll_no, name\} \rightarrow$ नाव हे क्षुल्लक कार्यात्मक अवलंबित्व आहे, कारण नाव हे निर्धारक संच $\{roll_no, name\}$ चा उपसंच आहे.

2. गैर-क्षुल्लक कार्यात्मक अवलंबित्व (Non-Trivial Functional Dependency)

अवलंबित्व (Dependent) हा निर्धारकाचा (Determinant) उपसंच (subset) नसतो.

जर $X \rightarrow Y$ आणि Y हा X चा उपसंच नसेल. तर त्याला नॉन-ट्रिव्हियल फंक्शनल डिपेंडेंसी म्हणतात.

उदाहरणार्थ: विद्यार्थी (Student) टेबल

roll_no	name	age
42	abc	17
43	pqr	18
44	xyz	18

येथे, $roll_no \rightarrow$ नाव हे एक क्षुल्लक कार्यात्मक अवलंबित्व आहे, कारण अवलंबून नाव निर्धारक $roll_no$ चा उपसंच नाही.

3. बहुमूल्य कार्यात्मक अवलंबित्व (Multivalued Functional Dependency)

बहुमूल्य अवलंबित्वामध्ये किमान दोन विशेषता असतात जे तिसऱ्या विशेषतावर अवलंबून असतात.

उदाहरणार्थ:

बाइक उत्पादक कंपनीचा विचार करा, जी दरवर्षी प्रत्येक मॉडेलमध्ये दोन रंग (लाल आणि काळा) तयार करते.

bike_model	manuf_year	color
M1001	2007	Red
M1001	2007	Black
M2012	2008	Red
M2012	2008	Black
M2222	2009	Red
M2222	2009	Black

येथे स्तंभ $manuf_year$ आणि रंग स्वतंत्र आहेत आणि $bike_model$ वर अवलंबून आहेत.

हे अवलंबित्व खालीलप्रमाणे दर्शविले जाऊ शकते:

bike_model → → **manuf_year**

4. संक्रमणात्मक कार्यात्मक अवलंबित्व अवलंबित्व (Transitive Functional Dependency)

आश्रित (dependent set) हे अप्रत्यक्षपणे (indirectly) निर्धारकावर (Determinant) अवलंबून असते.

जर $X \rightarrow Y$ & $Y \rightarrow Z$ असेल, तर $X \rightarrow Z$. हे एक संक्रमणात्मक कार्यात्मक अवलंबित्व आहे.

5. पूर्णपणे कार्यात्मक अवलंबित्व (Fully Functional Dependency)

एक गुणधर्म किंवा गुणधर्मांचा संच विशिष्टपणे दुसरी विशेषता किंवा गुणधर्मांचा संच निर्धारित करतो.

जर नातेसंबंध R मध्ये X, Y, Z हे अवलंबन $X \rightarrow Y$ आणि $X \rightarrow Z$ सह गुणविशेष असतील तर त्या अवलंबित्व पूर्णपणे कार्यशील आहेत असे सांगते.

2. 5. 2. सामान्य फॉर्म (Normal Forms):

पहिला सामान्य फॉर्म (1NF):

1NF मध्ये, प्रत्येक टेबल सेलमध्ये फक्त एक मूल्य (Single Atomic Value) असावे.

दुसरा सामान्य फॉर्म (2NF):

एखादे टेबल जर पहिल्या सामान्य स्वरूपात असेल तरच ते दुसऱ्या सामान्य स्वरूपात असू शकते. 2NF मध्ये, प्रत्येक नॉन-की विशेषता (Non-key attribute) प्राथमिक कीवर (Primary Key) अवलंबून असते.

तिसरा सामान्य फॉर्म (3NF):

जर टेबल प्रथम आणि द्वितीय सामान्य फॉर्ममध्ये (1NF and 2NF) आहे आणि ज्यामध्ये कोणतीही नॉन-प्राइमरी-की विशेषता (Non-prime attributes) प्राइमरी की वर संक्रमणात्मकपणे अवलंबून नाही (No Transitive Dependency), तर ते टेबल तृतीय सामान्य फॉर्म (3NF) मध्ये असते.

उदाहरणार्थ:

प्रश्न - दिलेली माहिती तृतीय सामान्य फॉर्ममध्ये सामान्य करा.

(Normalize this data to Third Normal Form)

कॉलेज आपल्या लेक्चरर्सच्या विषय क्षेत्र कौशल्यांचा तपशील ठेवते. या तपशीलांमध्ये Lecturer Number, Lecturer Name, Lecturer Grade, Department Code, Department Name, Subject Code, Subject Name, Subject Level हे समाविष्ट आहे. असे गृहीत धरा की प्रत्येक व्याख्याता अनेक विषय शिकवू शकतो परंतु एकापेक्षा जास्त विभागांचा असू शकत नाही. Subject Code, Subject Name and Subject Level ह्या फिल्डची पुनरावृत्ती होत आहे.

उत्तर:

असामान्य फॉर्म (Un-normalized form) UNF

Table 1:

(Lecturer Number (Primary Key), Lecturer Name, Lecturer Grade, Department Code, Department Name, Subject Code, Subject Name, Subject Level)

पहिला सामान्य फॉर्म (1NF):

Table 1:

(Lecturer Number (Primary Key), Lecturer Name, Lecturer Grade, Department Code, Department Name)

Table 2:

(Lecturer Number (Foreign Key), Subject Code (Primary Key), Subject Name, Subject Level)

दुसरा सामान्य फॉर्म (2NF):

Table 1:

(Lecturer Number (Primary Key), Lecturer Name, Lecturer Grade, Department Code, Department Name)

Table 2:

(Lecturer Number (Foreign Key), Subject Code (Primary Key))

Table 3:

(Subject Code (Primary Key), Subject Name, Subject Level)

तिसरा सामान्य फॉर्म (3NF):

Table 1:

(Lecturer Number (Primary Key), Lecturer Name, Lecturer Grade)

Table 2:

(Department Code (Primary Key), Department Name)

Table 3:

(Subject Code (Primary Key), Subject Name, Subject Level)

Table 4:

(Lecturer Number (Foreign Key), Subject Code (Primary Key), Department Code (Foreign Key))

References:

1. <https://nptel.ac.in/courses/106105175>
2. <https://www.w3schools.com/sql/>
3. <https://www.tutorialspoint.com/sql/index.htm>
4. Database System Concepts by Henry F. Korth McGraw Hill Education

युनिट - 3

इंटरॅक्टिव्ह SQL अँड परफॉर्मन्स ट्युनिंग

(Interactive SQL and Performance Tuning)

विषय निष्पत्ती (Course Outcome): एसक्यूएल वापरून डाटाबेस व्यवस्थापित करणे.

घटक निष्पत्ती (Theory Learning Outcome):

1. डीडीएल(DDL), डीएमएल(DML), डीसीएल(DCL) आणि टीसीएल (TCL) वापरून एसक्यूएल क्वेरी लिहा.
2. संबंधांमध्ये सामील होण्यासाठी एसक्यूएल क्वेरी लिहा.
3. डाटा ऑर्डर आणि गटबद्ध करण्यासाठी एसक्यूएल क्वेरी लिहा.
4. एसक्यूएलमध्ये ऑपरेटरच्या विविध वर्गांचा वापर करा.
5. परफॉर्मन्स ट्विनिंगसाठी स्कीमा ऑब्जेक्ट्स तयार करा.

3.1.1 एसक्यूएल (SQL)

- एसक्यूएल(स्ट्रक्चर्ड क्वेरी लॅंग्वेज) (SQL(Structured Query Language)) ही एएनएसआई (ANSI) (अमेरिकन नॅशनल स्टॅंडर्ड्स इन्स्टिट्यूट) ने डाटाबेस सिस्टमशी संवाद साधण्यासाठी (एॅक्सेस करणे आणि हाताळण्यासाठी) शिफारस केलेली मानक संगणक भाषा आहे. म्हणून, एसक्यूएल ही डाटाबेस गेटवे भाषा म्हणून देखील ओळखले जाते.
- एसक्यूएलचा (SQL) वापर एमएस एसक्यूएल सर्व्हर (MS SQL Server), ओरॅकल (Oracle), सायबेस (Sybase), एमएस एॅक्सेस (MS Access) इत्यादी डाटाबेस सॉफ्टवेअरद्वारे केला जातो.
- एसक्यूएलचा (SQL) ची पहिली आवृत्ती आईबीएम (IBM) मध्ये डोनाल्ड डी. चेंबरलिन आणि रेमंड एफ. बॉइस यांनी 1970 च्या दशकाच्या सुरुवातीला विकसित केली होती. ही आवृत्ती, ज्याला सुरुवातीला सिक्वेल (SEQUEL) म्हटले जायचे, डाटा हाताळण्यासाठी आणि पुनर्प्राप्त करण्यासाठी डिझाइन केले होती
- एसक्यूएलचा (SQL) ही एक नॉन-प्रोसिजरल (Non-procedural) क्वेरी भाषा आहे.

3.1.2 डाटा टाइप्स (Data Types):

- डाटा टाइप्स नमूद करते की कॉलम मध्ये कोणत्या प्रकारच्या डाटा साठवता येतो.
- डाटा टाइप्सच्या कॉलममध्ये फक्त योग्य प्रकारच्या डाटाला साठवण्याची परवानगी आहे याची खात्री करून त्रुटी तपासण्याचे साधन प्रदान करते.

3.1.2 बेसिक डाटा टाइप्स (Basic data types)

1. कॅरेक्टर(Character)

- कॅरेक्टर डाटा टाइप्स निश्चित लांबीचा कॅरेक्टर डाटा साठवण्यासाठी वापरले जाते.
- कॅरेक्टर डाटा टाइप्सची कमाल क्षमता (Maximum size) 2000 बाइट्स आहे.
- कॅरेक्टर डाटा टाइप्स बायडीफॉल्ट 1 बाइट साठवण्यासाठी सक्षम आहे.
- कॅरेक्टर डाटा टाइप्सचा वापर करण्यासाठी कॅर(char) कीवर्ड वापरला जातो
- **उदाहरण:** name char(10);

2. व्हारकॅर/ व्हारकॅर2 (varchar/varchar2)

- व्हारकॅरडाटा टाइप्स परिवर्तनशील (variable-length) लांबीच्या कॅरेक्टर डाटा साठवण्यासाठी वापरले जाते.
- व्हारकॅरडाटा डाटा टाइप्सची कमाल क्षमता (Maximum size) 4000 बाइट्स आहे.
- व्हारकॅरडाटा डाटा टाइप्स मध्ये किमान डाटा (Minimum size) साठवण्यासाठी क्षमता नाही.

- व्हारकॅरडाटा डाटा टाइप्स मध्ये डाटा साठवण्यासाठी varchar(size), varchar2(size) म्हणून निर्दिष्ट करणे आवश्यक आहे.
- व्हारकॅरडाटा डाटा टाइप्सचा वापर करण्यासाठी कॅर (varchar/varchar2) कीवर्ड वापरला जातो

उदाहरण:

```
name varchar(15);
emp_name varchar2(20);
```

3. संख्या डाटा प्रकार (Number Data Types)

1. पूर्णांक: columnname NUMBER(precision)

उदाहरण

```
rollno number(5);
```

2. निश्चित बिंदू: columnname NUMBER([precision],[scale])

उदाहरण

```
price number(5, 2);
```

3. फ्लोटिंग पॉइंट: columnname NUMBER

उदाहरण

```
no number;
```

4. डेट(Date)

- डेट डाटा टाइप्स वास्तविक तारीख आणि वेळ मूल्ये साठवण्यासाठी वापरले जाते.
- डेट डाटा टाइप्सचे डीफॉल्ट स्वरूप 'DD-MON-RR' हे आहे.

उदाहरण

```
dob DATE;
```

5. एलओबी (लार्ज ऑब्जेक्ट) (Large Objects(LOBs))

सर्व एलओबी (LOB) कॉलममध्ये डाटा टेबलमध्ये जात नाही. मूळ तर टेबलमध्ये डाटाबेस लोकेटर (पॉइंटर्स) स्टोअर करते, जे डाटाबेसमध्ये स्थान सूचित करते जे डाटाने एलओबी ऑब्जेक्ट्स ठेवण्यासाठी तयार केले आहे.

1. सीएलओबी(कॅरेक्टर लार्ज ऑब्जेक्ट्स)(Character Large Objects (CLOB)): कॅरेक्टर लार्ज ऑब्जेक्ट्स डाटा टाइप्सची कमाल क्षमता(Maximum size): 4 जिबी(4GB) आहे.
2. एनसीएलओबी (मल्टीबाइट कॅरेक्टर लार्ज ऑब्जेक्ट्स) (Multibyte Character Large Objects (NCLOB)): मल्टीबाइट कॅरेक्टर लार्ज ऑब्जेक्ट्स हे कॅरेक्टर लार्ज ऑब्जेक्ट्स प्रमाणेच आहे, परंतु मल्टीबाइट कॅरेक्टर सेटसाठी वापरले जाते.
3. बीएलओबी (बायनरी लार्ज ऑब्जेक्ट्स)(Binary Large Objects (BLOB)): बायनरी लार्ज ऑब्जेक्ट्स डाटा टाइप्सची कमाल क्षमता(Maximum size) 4 जिबी (4GB) आहे.

6. बीफाईल(BFILE): बीफाईल ही डाटा टाइप्स बायनरी ऑपरेटिंग सिस्टम फाइलकडे निर्देश करण्यासाठी वापरले जाते.

3.1.3 एसक्यूएल (SQL) कमांड खालील श्रेणींमध्ये विभागल्या आहेत:

1. डाटा डेफिनिशन लॅंग्वेज (Data Definition Language(DDL)): डीडील (DDL) , ज्याचा अर्थ डाटा डेफिनिशन लॅंग्वेज आहे, डाटाबेस संरचना परिभाषित आणि सुधारित करण्यासाठी वापरल्या जाणाऱ्या एसक्यूएल (स्ट्रक्चर्ड क्वेरी लॅंग्वेज) कमांडचा एक उपसंच आहे. या कमांड्सचा उपयोग डाटाबेस ऑब्जेक्ट्स जसे की टेबल्स (tables), व्ह्यू (views), सीक्वेंस (sequences), सिनोनिम्स (synonyms), इंडेक्स (indexes) आणि स्कीमा (schema) तयार करण्यासाठी, बदलण्यासाठी आणि हटवण्यासाठी केला जातो.

खालील नमूद केलेल्या डाटा डेफिनिशन लॅंग्वेज च्या कमांडस आहेत

1. क्रिएट (CREATE): क्रिएट (CREATE) कमांडचा वापर कोणताही डाटाबेस ऑब्जेक्ट तयार करण्यासाठी केला जातो.

वाक्यरचना (Syntax)

```
CREATE TABLE tablename
(columnname1 data_type (size) [constraints],
(columnname2 data_type (size) [constraints],.....)
```

उदाहरण

```
Create table mystuds
(rollno number(5),
Name varchar2(25),
Total number(5,2) default 0);
```

--या उदाहरणात आपण टेबल ऑब्जेक्ट तयार केला आहे.

2. ऑल्टर (ALTER): आधीच अस्तित्वात असलेली डाटाबेस संरचना बदलण्यासाठी ऑल्टर(ALTER) कमांडचा वापर केला जातो.

वाक्यरचना (Syntax)

```
ALTER TABLE name_of_table ADD column_name column_definition;
```

उदाहरण

```
ALTER TABLE Student ADD Father's_Name Varchar(60);
```

--या उदाहरणात आपण टेबल ऑब्जेक्टच्या संरचने मध्ये बदल केला आहे.

3. ड्राप (DROP): डाटाबेसमधून डाटाबेस ऑब्जेक्ट्स हटवण्यासाठी/काढण्यासाठी ड्राप (DROP) कमांड वापरला जातो.

वाक्यरचना (Syntax)

```
DROP TABLE Table_Name;
```

उदाहरण

```
DROP TABLE Student;
```

या उदाहरणात आपण टेबल ऑब्जेक्ट डाटाबेसमधून काढला आहे.

4. ट्रंकेट (TRUNCATE): ट्रंकेट कमांडचा वापर ऑब्जेक्ट्स मधील सर्व रेकॉर्ड हटवण्यासाठी किंवा काढून टाकण्यासाठी केला जातो. ही कमांड टेबल रेकॉर्ड्स साठवण्यासाठी वाटप केलेली जागा देखील काढून टाकते.

वाक्यरचना (Syntax)

```
TRUNCATE TABLE Table_Name;
```

उदाहरण

```
TRUNCATE TABLE Student;
```

5. रिनेम (RENAME): डाटाबेस ऑब्जेक्ट्स नाव बदलण्यासाठी रिनेम (RENAME) कमांडचा वापर केला जातो.

वाक्यरचना (Syntax)

```
RENAME TABLE Old_Table_Name TO New_Table_Name;
```

उदाहरण

```
RENAME TABLE Student TO Student_Info ;
```

6. कमेंट (COMMENT): डाटा डिक्शनरीमध्ये टिप्पण्या जोडाण्यासाठी कमेंट (COMMENT) कमांडचा वापर केला जातो.

वाक्यरचना (Syntax)

```
COMMENT 'comment_text' ON TABLE table_name;
```

उदाहरण

COMMENT ON TABLE Student IS 'This table contains details of all students.';

डीडीएल (DDL) कमांड्समुळे इम्प्लिसिट कमिट(ऑटो सेव्ह) होते. (त्या बिंदूपर्यंतचे अनकमिटेड डीएमएल डाटाबेसमध्ये सेव्ह केले जातात आणि व्यवहार संपतो)

2. डाटा मॅनिप्युलेशन लॅंग्वेज (Data Manipulation Language(DML)): डाटा मॅनिप्युलेशन लॅंग्वेज (डीएमएल) हा डाटाबेसमध्ये डाटा जोडणे (इन्सर्ट करणे), हटवणे आणि सुधारित करणे (अपडेट करणे) यासाठी वापरल्या जाणाऱ्या एसक्यूएल कमांडचा एक उपसंच आहे. डाटाबेसच्या टेबलमधील डाटा व्यवस्थापित करण्यासाठी डीएमएल कमांड्स महत्त्वपूर्ण आहेत.

1. इन्सर्ट (INSERT): इन्सर्ट (INSERT) कमांड वापरकर्त्यांना डाटाबेस ऑब्जेक्टमध्ये डाटा इन्सर्ट करण्याची परवानगी देते.

वाक्यरचना (Syntax)

INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);

उदाहरण

INSERT INTO Student (S_id, Stu_Name, S_Marks, S_Age) VALUES (111, Amol, 92, 19);

2. अपडेट (UPDATE): अपडेट (UPDATE) कमांड वापरकर्त्यांना डाटाबेस टेबलमधील विद्यमान डाटा अपडेट किंवा सुधारित करण्याची परवानगी देते.

वाक्यरचना (Syntax)

UPDATE table_name SET column1 = value1, column2 = value2 WHERE condition;

उदाहरण

UPDATE Student SET Marks = 90, Age = 18 WHERE Id = 111;

3. डिलीट (DELETE): डिलीट (DELETE) कमांड वापरकर्त्यांना डाटाबेस मधून एक किंवा अनेक विद्यमान रेकॉर्ड काढण्याची परवानगी देते. डिलीट (DELETE) कमांड डाटाबेसमधून संग्रहित डाटा कायमचा हटवत नाही. टेबलमधून विशिष्ट पंक्ती निवडण्यासाठी आम्ही डिलीट(DELETE) कमांडसह वेयर(WHERE) क्लॉज वापरतो.

वाक्यरचना (Syntax)

DELETE FROM Table_Name WHERE condition;

उदाहरण

DELETE FROM Student WHERE Marks > 75 ;

3. डाटा कंट्रोल लॅंग्वेज (Data Control Language (DCL)): डाटा कंट्रोल लॅंग्वेज (डीसीएल) हा डाटाबेसमधील डाटावर प्रवेश नियंत्रित करण्यासाठी वापरल्या जाणाऱ्या एसक्यूएल कमांडचा उपसंच आहे. विशेषतः बहु-वापरकर्ता डाटाबेस वातावरणात सुरक्षितता आणि योग्य डाटा व्यवस्थापन सुनिश्चित करण्यासाठी डीसीएलच्या कमांड्स महत्वाच्या आहेत.

1. ग्रॅन्ट (GRANT) : विशिष्ट डाटाबेस ऑब्जेक्ट्स, क्रिया किंवा फंक्शन्समध्ये प्रवेशास अनुमती देऊन, वापरकर्ता खात्यासाठी नवीन विशेषाधिकार नियुक्त करते.

वाक्यरचना (Syntax)

GRANT privilege_type [(column_list)] ON [object_type] object_name TO user [WITH GRANT OPTION];

2. रिवोक(REVOKE): वापरकर्ता खात्यातून पूर्वी मंजूर केलेले विशेषाधिकार काढून टाकते, विशिष्ट डाटाबेस ऑब्जेक्ट्स किंवा कृतींवरील त्यांचा प्रवेश काढून टाकते.

वाक्यरचना (Syntax)

REVOKE [GRANT OPTION FOR] privilege_type [(column_list)] ON [object_type]
object_name FROM user [CASCADE];

4. ट्रान्झॅक्शन कंट्रोल लॅंग्वेज (Transaction control Language (TCL)): ट्रान्झॅक्शन कंट्रोल लॅंग्वेज (टीसीएल) हा डाटाबेसमधील व्यवहार व्यवस्थापित करण्यासाठी वापरल्या जाणाऱ्या एसक्यूएल कमांडचा उपसंच आहे. डाटाची अखंडता आणि सातत्य राखण्यासाठी व्यवहार महत्त्वाचे आहेत. ते एकाधिक डाटाबेस ऑपरेशन्स कार्याचे एक युनिट म्हणून कार्यान्वित करण्याची परवानगी देतात, जे एकतर पूर्णपणे यशस्वी होतात किंवा अयशस्वी होतात.

1. कमिट (COMMIT): कमिट(COMMIT) कमांडचा वापर सध्याच्या व्यवहारात केलेले सर्व बदल कायमस्वरूपी सेव्ह करण्यासाठी केला जातो.

वाक्यरचना (Syntax)

COMMIT;

2. रोलबैक (ROLLBACK): रोलबैक(ROLLBACK) कमांडचा वापर सध्याच्या व्यवहारात केलेले बदल पूर्ववत करण्यासाठी केला जातो.

वाक्यरचना (Syntax)

ROLLBACK;

3. सवेपॉईंट (SAVEPOINT): सवेपॉईंट (SAVEPOINT) कमांड ट्रान्झॅक्शनमध्ये पॉइंट तयार करते ज्यावर तुम्ही नंतर रोल बॅक करू शकता. हे आंशिक रोलबॅक आणि अधिक जटिल व्यवहार नियंत्रणास अनुमती देते.

वाक्यरचना (Syntax)

SAVEPOINT savepoint_name;

3.2.1 क्लॉज (CLAUSES): एसक्यूएल क्लॉज हे स्ट्रक्चर्ड क्वेरी लॅंग्वेजमधील मूलभूत घटक आहेत ज्याचा वापर डाटा पुनर्प्राप्ती, हाताळणी आणि रिलेशनल डाटाबेसमधील संस्थेसाठी विशिष्ट निकष परिभाषित करण्यासाठी केला जातो. ही कलमे एसक्यूएल क्वेरीचे आवश्यक भाग बनवतात, ज्यामुळे वापरकर्त्यांना डाटाबेस रेकॉर्डशी संवाद साधता येतो आणि ते प्रभावीपणे हाताळता येते.

क्लॉजचे (Clauses) विविध प्रकार

1. वेयर क्लॉज (Where Clauses): विशिष्ट परिस्थितींवर आधारित रेकॉर्ड फिल्टर करण्यासाठी सिलेक्ट सह वेयर क्लॉज वापरला जातो.

वाक्यरचना (Syntax)

SELECT column1, column2, ... FROM table_name WHERE condition;

उदाहरण

SELECT * FROM Student WHERE department = 'CO';

2. ऑर्डर बाय क्लॉज (Order by Clauses): ऑर्डर बाय क्लॉज परिणाम-सेटला चढत्या किंवा उतरत्या क्रमाने वर्गीकरण करतो. हे डीफॉल्टनुसार चढत्या क्रमाने रेकॉर्डची क्रमवारी लावते. डीइएससी(DESC) कीवर्डचा वापर नोंदी उतरत्या क्रमाने क्रमवारी लावण्यासाठी केला जातो.

वाक्यरचना (Syntax)

SELECT column1, column2
FROM table_name
WHERE condition
ORDER BY column1, column2... ASC|DESC;

उदाहरण

SELECT *
FROM Student

ORDER BY NAME;

3. ग्रुप बाय क्लॉज (Group by Clauses): एसक्यूएल मधील ग्रुप बाय स्टेटमेंटचा वापर समान डाटा गटांमध्ये मांडण्यासाठी केला जातो. ग्रुप बाय स्टेटमेंट सिलेक्ट स्टेटमेंटसह वापरले जाते. ग्रुप बाय स्टेटमेंट हे सिलेक्ट स्टेटमेंटमधील वेयर क्लॉजचे अनुसरण करते आणि ऑर्डर बाय क्लॉजच्या आधी येते.

ग्रुप बाय स्टेटमेंट विधान एकत्रीकरण कार्यासह वापरले जाते.

वाक्यरचना (Syntax)

```
SELECT column1, function_name(column2)
FROM table_name
WHERE condition
GROUP BY column1, column2
ORDER BY column1, column2;
```

उदाहरण

```
SELECT SUBJECT, YEAR, Count(*)
FROM Student
GROUP BY SUBJECT, YEAR;
```

4. हॅविंग क्लॉज (Having Clauses): एग्रीगेट फंक्शन्स आणि ग्रुपिंगवर आधारित केरी परिणाम फिल्टर करण्यास अनुमती देण्यासाठी एसक्यूएल मध्ये हॅविंग क्लॉज सादर करण्यात आला आहे, जो वैयक्तिक पंक्ती फिल्टर करण्यासाठी वापरला जाणारा वेयर क्लॉज वापरून साध्य करता येत नाही.

वाक्यरचना (Syntax)

```
SELECT column1, column2
FROM table_name
WHERE conditions
GROUP BY column1, column2
HAVING conditions
ORDER BY column1, column2;
```

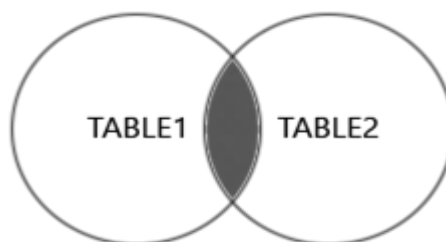
उदाहरण

```
SELECT student, percentage
FROM Student
GROUP BY student, percentage
HAVING percentage > 88;
```

3.2.2 जॉईन (Join): एसक्यूएल जॉईन ऑपरेशनमध्ये दोन किंवा अधिक टेबलमधील डाटा किंवा रो एकत्रित केल्या जातात आणि त्यांच्यामधील समान फील्डवर आधारित असतात.

एसक्यूएल जॉईनचे प्रकार (Types of SQL Join)

1. इनर जॉईन/सिंपल जॉईन (Inner Join / Simple Join): दोन्ही टेबलमधील कॉमन रो निवडण्यासाठी इनर जॉईन वापरला जातो. आपण खालीलप्रमाणे वेन आकृतीद्वारे इनर जॉईन दर्शवू शकतो:



आकृती 3.1 इनर जॉईन/सिंपल जॉईन (Inner Join / Simple Join)

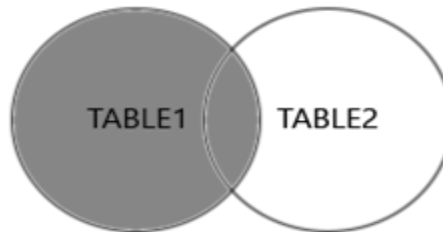
वाक्यरचना (Syntax)

```
Select column_1, column_2, column_3 FROM table_1 INNER JOIN table_2 ON
table_1.column = table_2.column;
```

उदाहरण

```
Select ID, Name, TeacherName
FROM Student INNER JOIN Teacher ON Student.TID = Teacher.TID;
```

2. लेफ्ट जॉईन (Left Join): लेफ्ट जॉईन मध्ये डाव्या टेबलमधील सर्व रेकॉर्ड आणि उजव्या टेबलमधील जुळलेले रेकॉर्ड दाखवले जातात. आपण खालीलप्रमाणे वेन आकृतीद्वारे लेफ्ट जॉईन दर्शवू शकतो:



आकृती 3.2 लेफ्ट जॉईन (Left Join)

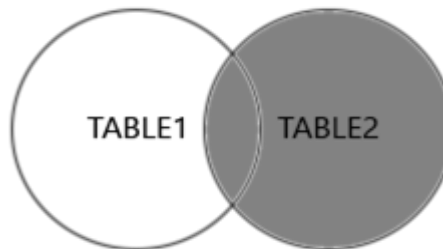
वाक्यरचना (Syntax)

```
Select column_1, column_2, column(s) FROM table_1 LEFT JOIN table_2 ON
table_1.column_name = table_2.column_name;
```

उदाहरण

```
Select ID, PName, CName, CAddress, Amount
FROM Product_Detail LEFT JOIN Customer_Detail
ON Product_Details.ID = Customer_Details.PID;
```

3. राइट जॉईन (Right Join): राइट जॉईन मध्ये उजव्या टेबलमधील सर्व रेकॉर्ड आणि डाव्या टेबलमधील जुळलेले रेकॉर्ड दाखवले जातात. आपण खालीलप्रमाणे वेन आकृतीद्वारे राइट जॉईन दर्शवू शकतो:



आकृती 3.3 राइट जॉईन (Right Join)

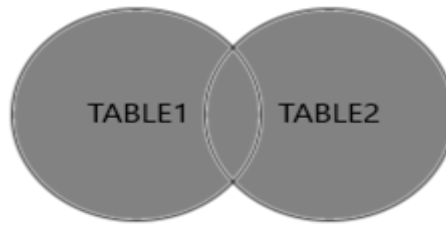
वाक्यरचना (Syntax)

```
Select column_1, column_2, column(s) FROM table_1 RIGHT JOIN table_2 ON
table_1.column_name = table_2.column_name;
```

उदाहरण

```
Select ID, PName, CName, CAddress, Amount
FROM Product_Detail RIGHT JOIN Customer_Detail
ON Product_Detail.ID = Customer_Detail.PID;
```

4. फुल आऊटर जॉईन (Full Outer Join): फुल आऊटर जॉईन मध्ये उजव्या टेबलमधील किंवा डाव्या टेबलमधील सर्व मॅच रेकॉर्ड दाखवले जातात. आपण खालीलप्रमाणे वेन आकृतीद्वारे राइट जॉईन दर्शवू शकतो:



आकृती 3. 4 फुल आऊटर जॉईन (Full Outer Join)

वाक्यरचना (Syntax)

```
Select column_1, column_2, column(s) FROM table_1 FULL JOIN table_2 ON
table_1.column_name = table_2.column_name;
```

उदाहरण

```
Select ID, PName, CName, CAddress, Amount
FROM Product_Detail FULL JOIN Customer_Detail
ON Product_Details.ID = Customer_Detail.ProductID;
```

5. क्रॉस जॉईन (Cross Join): क्रॉस जॉईन हे कार्टेशियन प्रॉडक्ट तयार करते, म्हणजेच टेबल X मधील प्रत्येक रो टेबल Y मधील प्रत्येक रोशी मॅप केली जाते. जर जॉईन होण्याची अट वगळली असेल किंवा जॉईन अट नेहमी सत्य असेल तर क्रॉस जॉईन वापरला जातो.

वाक्यरचना(Syntax)

```
Select * from table_1 cross join table_2;
```

उदाहरण

```
Select * FROM Product_Detail cross join Customer_Detail;
```

6. सेल्फ जॉईन (Self Join): सेल्फ जॉईन म्हणजे एकाच टेबलमधील एक रो दुसऱ्या रोशी जॉईन करण्यासाठी वापरला जातो. सेल्फ जॉईन मध्ये किमान एक टेबल तात्पुरता रीनेम करावा लागतो.

वाक्यरचना (Syntax)

```
Select column1, column2, column(s) FROM table1 t1, table2 t2 WHERE condition;
```

उदाहरण

```
Select p1.ID, p1.PName
FROM Product_Detail p1, Product_Detail p2
WHERE p1.AMOUNT < p2.AMOUNT;
```

3.2.3 नेस्टेड क्वेरी (Nested Query): एसक्यूएल मधील नेस्टेड क्वेरीमध्ये दुसऱ्या क्वेरीमध्ये एक क्वेरी असते. आउटर क्वेरी इनर क्वेरीचा रिजल्ट वापरेल.

वाक्यरचना (Syntax)

```
SELECT column1,column2,..., columnN FROM table1 WHERE column1 IN ( SELECT
column1 FROM table2 WHERE condition );
```

उदाहरण

```
SELECT SName
FROM Student WHERE SID IN (SELECT SID FROM Grade
WHERE Subject = 'Mathematics' AND Score > 95);
```

3.3 ऑपरेटर (Operators): एसक्यूएल मधील ऑपरेटर ही सिंबॉल आहेत जी आपल्याला ऑपरेंडवर विशिष्ट मॅथेमॅटिकल आणि लॉजिकल कंप्यूटेशन करण्यास मदत करतात. ऑपरेटर एकतर यूनरी किंवा बायनरी असू शकतो.

1. अरीथमेटिक ऑपरेटर (Arithmetic Operators): एसक्यूएल मधील अरीथमेटिकल ऑपरेटर क्वेरीमधील मॅथेमेटिकल ऑपरेशन्स परफॉर्म करण्यासाठी वापरले जातात. काही अरीथमेटिक ऑपरेटर खाली नमुद केले आहेत:

ऑपरेटर	ऑपरेशन्स	डिस्क्रिप्शन (Description)
+	अॅडिशन (Addition)	अॅडिशन ऑपरेशन हे डाटा व्हॅल्यूचे अॅडिशन करण्यासाठी वापरले जाते. SELECT item, amount, amount+100 AS total_amount FROM Orders;
-	सबट्रॅक्शन (Subtraction)	सबट्रॅक्शन ऑपरेटर हे डाटा व्हॅल्यूचे सबट्रॅक्शन करण्यासाठी वापरले जाते. SELECT item, amount, amount-20 AS offer_price FROM Orders;
*	मल्टिप्लिकेशन (Multiplication)	मल्टिप्लिकेशन ऑपरेटर डाटा व्हॅल्यूचे मल्टिप्लिकेशन करण्यासाठी वापरले जाते.
/	डीव्हिजन (Division)	डीव्हिजन ऑपरेटर डाव्या हाताच्या ऑपरेंडला उजव्या हाताच्या ऑपरेंडने डीव्हाइड करण्यासाठी वापरले जाते.
%	मॉड्यूलस (Modulus)	मॉड्यूलस ऑपरेटर डाव्या हाताच्या ऑपरेंडला उजव्या हाताने ऑपरेंड डीव्हाइड करते आणि रिमाइंडर रिझल्ट रिटन देते.

2. कम्पॅरिझन ऑपरेटर (Comparison Operators): एसक्यूएलमधील कम्पॅरिझन ऑपरेटर्सचा वापर एका एक्सप्रेशन व्हॅल्यूची इतर एक्सप्रेशन व्हॅल्यूशी तुलना करण्यासाठी वापरले जाते. एसक्यूएल विविध प्रकारच्या कम्पॅरिझन ऑपरेटरला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

ऑपरेटर	ऑपरेशन्स	डिस्क्रिप्शन (Description)
=	इक्वल टू (Equal to)	इक्वल टू ऑपरेटर दोन्ही ऑपरेंडचे व्हॅल्यू इक्वल आहे का ते तपासते, जर व्हॅल्यू इक्वल असेल, तर रिझल्ट टू रिटन करते.
>	ग्रेटर दॅन (Greater than)	ग्रेटर दॅन ऑपरेटर डाव्या हाताच्या ऑपरेंडचे व्हॅल्यू उजव्या हाताच्या ऑपरेंडपेक्षा ग्रेटर आहे की नाही ते तपासते.
<	लेस दॅन (Less than)	लेस दॅन ऑपरेटर डाव्या हाताच्या ऑपरेंडचे व्हॅल्यू उजव्या हाताच्या ऑपरेंडच्या व्हॅल्यूपेक्षा कमी असल्यास रिझल्ट टू रिटन करते.
>=	ग्रेटर दॅन ऑर इक्वल टू (Greater than or equal to)	ग्रेटर दॅन ऑर इक्वल टू ऑपरेटर डाव्या हाताच्या ऑपरेंडचे व्हॅल्यू उजव्या हाताच्या ऑपरेंडच्या व्हॅल्यू पेक्षा मोठे किंवा समान आहे का ते तपासते, जर असेल, तर रिझल्ट टू रिटन करते.
<=	लेस दॅन ऑर इक्वल टू (Less than or equal to)	लेस दॅन ऑर इक्वल टू ऑपरेटर डाव्या हाताच्या ऑपरेटरचे व्हॅल्यू उजव्या हाताच्या ऑपरेंडपेक्षा कमी किंवा समान आहे का ते तपासते, जर असेल, तर रिझल्ट टू रिटन करते.
<>	नॉट इक्वल टू (Not equal to)	नॉट इक्वल टू ऑपरेटर ऑपरेटरच्या दोन्ही बाजूची व्हॅल्यू समान आहेत की नाही ते तपासते. व्हॅल्यू समान नसल्यास रिझल्ट टू रिटन करते.

3. लॉजिकल ऑपरेटर (Logical Operators): एसक्यूएलमधील लॉजिकल ऑपरेटर बुलियन ऑपरेशन्स करतात, जे टू आणि फाल्स असे दोन परिणाम देतात. एसक्यूएल विविध प्रकारच्या लॉजिकल ऑपरेटरला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

ऑपरेटर	ऑपरेशन्स	डिस्क्रिप्शन (Description)
AND	अँड (AND)	अँड ने अप्लाय केलेल्या सर्व कंडिशन टू असल्यास रेकॉर्ड रिटन करते.
OR	ओर (OR)	ओर दोन बुलियन एक्सप्रेशन कम्पेअर करते आणि दोघांपैकी एकही टू असेल तर रेकॉर्ड रिटन करते.
NOT	नॉट (NOT)	नॉट ऑपरेटर एक बुलियन आर्ग्युमेंट इनपुट म्हणून घेते आणि त्याची व्हॅल्यू निगेटिव्ह असेल तर पॉझिटिव्ह मध्ये कन्व्हर्ट करते किंवा पॉझिटिव्ह व्हॅल्यू निगेटिव्ह व्हॅल्यू मध्ये कन्व्हर्ट करते.

4. सेट ऑपरेटर (Set Operators): सेट ऑपरेटर हे विशेष प्रकारचे ऑपरेटर आहेत जे दोन क्वेरींचे परिणाम एकत्र करण्यासाठी वापरले जातात. एसक्यूएल विविध प्रकारच्या सेट ऑपरेटरला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

ऑपरेटर	ऑपरेशन्स	डिस्क्रिप्शन (Description)
UNION	युनियन (UNION)	युनियन ऑपरेशन दोन किंवा अधिक एसक्यूएल सिलेक्ट क्वेरींचे परिणाम एकत्र करण्यासाठी वापरले जाते. युनियन ऑपरेशनमध्ये, युनियन ऑपरेशन लागू होत असलेल्या दोन्ही टेबलमध्ये डाटाटाइप आणि कॉलम्सची संख्या समान असणे आवश्यक आहे. युनियन ऑपरेशन रिझल्ट मधील डुप्लिकेट रो काढून टाकते. वाक्यरचना(Syntax) SELECT column_name FROM table1 UNION SELECT column_name FROM table2;
Union All	युनियन ऑल (Union All)	युनियन ऑल ऑपरेशन युनियन ऑपरेशन प्रमाणेच आहे. युनियन ऑल ऑपरेशन डुप्लिकेशन काढल्याशिवाय आणि डाटाची क्रमवारी न लावता रिझल्ट परत करते. वाक्यरचना(Syntax) SELECT column_name FROM table1 UNION ALL SELECT column_name FROM table2;
Intersect	इंटरसेक्ट (Intersect)	इंटरसेक्ट ऑपरेशन हे दोन सिलेक्ट क्वेरींचे परिणाम एकत्र करण्यासाठी वापरले जाते. इंटरसेक्ट ऑपरेशन दोन्ही सिलेक्ट विधानांमधून इक्वल रो मिळवते. इंटरसेक्ट ऑपरेशनमध्ये, डाटाटाइप आणि कॉलम संख्या इक्वल असणे आवश्यक आहे. इंटरसेक्ट ऑपरेशन मध्ये कोणतेही डुप्लिकेट रो रिझल्ट मध्ये येत नाहीत आणि रिझल्ट बाय डिफॉल्ट चढत्या क्रमाने अरेंज केला जातो. वाक्यरचना(Syntax) SELECT column_name FROM table1 INTERSECT SELECT column_name FROM table2;
Minus	मायनस (Minus)	मायनस ऑपरेशन हे दोन सिलेक्ट क्वेरींचे परिणाम एकत्र करते. मायनस ऑपरेटरचा वापर पहिल्या क्वेरीमध्ये उपस्थित असलेल्या परंतु दुसऱ्या क्वेरीमध्ये अनुपस्थित असलेल्या पंक्ती प्रदर्शित करण्यासाठी केला जातो.

ऑपरेटर	ऑपरेशन्स	डिस्क्रिप्शन (Description)
		मायनस ऑपरेशन मध्ये कोणतेही डुप्लिकेट रो रिझल्ट मध्ये येत नाहीत आणि रिझल्ट बाय डिफॉल्ट चढत्या क्रमाने अरेंज केला जातो. वाक्यरचना(Syntax) SELECT column_name FROM table1 MINUS SELECT column_name FROM table2;

3.4 फंक्शन (Functions)

1. न्यूमरिक फंक्शन (Numeric Functions): एसक्यूएल विविध प्रकारच्या न्यूमरिक फंक्शनला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

फंक्शन (Function)	डिस्क्रिप्शन (Description)
एबीएस (ABS())	हे एका संख्येचे परिपूर्ण व्हॅल्यू मिळवते.
सिलिंग (CEILING())	संख्येपेक्षा मोठे किंवा समान असलेले सर्वात लहान पूर्णांक मूल्य मिळवते.
कॉस (COS())	रेडियनमध्ये कोसाइन व्हॅल्यू मिळवते.
कॉट (COT())	रेडियनमध्ये कोटॅजंट व्हॅल्यू मिळवते.
डिग्रीज (DEGREES())	रेडियन मधून अंशांमध्ये रूपांतरित केलेली अंकीय अभिव्यक्ती मिळवते.
एक्सपी (EXP())	उत्तीर्ण संख्यात्मक अभिव्यक्तीच्या बळावर वाढलेल्या नैसर्गिक लॉगरिदमचा (e) आधार मिळवते.
फ्लोअर (FLOOR())	संख्येपेक्षा कमी किंवा बरोबरीचे सर्वात मोठे पूर्णांक मूल्य मिळवते.
लॉग (LOG())	संख्येचा नैसर्गिक लॉगरिदम मिळवते.
लॉग10 (LOG10())	संख्येचा बेस 10 लॉगरिदम मिळवते.
पाय (PI())	पाय चे मूल्य मिळवते
पॉवर (POWER())	1ली संख्या 2-या संख्येच्या पॉवरवर वाढवते आणि निकाल मिळवते.
रेडियन (RADIANS())	अंशांमधून रेडियनमध्ये रूपांतरित केलेल्या उत्तीर्ण अभिव्यक्तीचे मूल्य मिळवते.
रँड (RAND())	0 आणि 1 मधील रँडम मूल्य मिळवते.

फंक्शन (Function)	डिस्क्रिप्शन (Description)
राऊंड (ROUND())	1ल्या संख्येला पूर्णांक किंवा 2-्या संख्येने दर्शविल्याप्रमाणे अनेक दशांश स्थानांवर पूर्ण करतो आणि परिणाम मिळवतो.
साईन (SIGN())	संख्येचे चिन्ह दाखवते, ते पॉझिटिव्ह, निगेटिव्ह किंवा शून्य आहे.
साईन (SIN())	रेडियनमध्ये दिलेल्या अंकीय अभिव्यक्तीची साइन मिळवते.
एसक्यूआरटी (SQRT())	संख्येचे वर्गमूळ मूल्य मिळवते.
टॅन (TAN())	रेडियनमध्ये स्पर्शिका मूल्य मिळवते.

2. डेट अँड टाईम फंक्शन (Date and Time Functions): एसक्यूएल विविध प्रकारच्या डेट अँड टाईम फंक्शनला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

फंक्शन (Function)	डिस्क्रिप्शन (Description)
नाऊ (NOW())	वर्तमान तारीख आणि वेळ मिळवते. SELECT NOW();
सिस डेट (SYSDATE)	सिस डेट फंक्शन तुमच्या डाटाबेसवर वर्तमान सिस्टम तारीख आणि वेळ देईल. SELECT Sysdate FROM Dual;
नेक्स्ट डे (NEXT_DAY())	दिलेल्या तारखेपेक्षा मोठा असलेला पहिला आठवड्याचा दिवस परत करण्यासाठी नेक्स्ट डे फंक्शन वापरले जाते. NEXT_DAY(DATE,WEEKDAY) DATE –तारीख व्हॅल्यू जे पुढील आठवड्याचा दिवस शोधण्यासाठी वापरले जाते. WEEKDAY - हा आठवड्याचा दिवस आहे जो आम्हाला परत करायचा आहे. उदाहरण SELECT NEXT_DAY(DATE '2024-11-06', 'MONDAY') FROM dual;
एँड मंथ (ADD_MONTHS())	एँड मंथ फंक्शन जोडलेल्या महिन्यांच्या संख्येसह तारीख मिळवते (तारीख अधिक पूर्णांक महिने). ADD_MONTHS(date, month) date – हे प्रारंभिक तारीख निर्दिष्ट करण्यासाठी वापरले जाते.

फंक्शन (Function)	डिस्क्रिप्शन (Description)
	month – सुरुवातीच्या तारखेला किती महिने जोडले जातील हे निर्दिष्ट करण्यासाठी याचा वापर केला जातो. उदाहरण SELECT ADD_MONTHS(DATE '2024-06-11', 3) FROM dual;
लास्ट डे (LAST_DAY())	दिलेल्या तारखेसाठी किंवा तारखेसाठी महिन्याचा शेवटचा दिवस जाणून घेण्यासाठी लास्ट डे फंक्शन वापरले जाऊ शकते. LAST_DAY(Date);
मंथ बिटवीन (MONTHS_BETWEEN())	मंथ बिटवीन फंक्शन दिलेल्या दोन तारखांमधील फरक जाणून घेण्यासाठी वापरला जातो. MONTHS_BETWEEN(date1, date2) date1 : फरक मोजण्यासाठी पहिली तारीख date2 : फरक मोजण्यासाठी दुसरी तारीख उदाहरण select MONTHS_BETWEEN('11-jun-2024','11-feb-2024') from dual;

3. स्ट्रिंग फंक्शन (String functions): एसक्यूएल विविध प्रकारच्या स्ट्रिंग फंक्शनला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

फंक्शन (Function)	डिस्क्रिप्शन (Description)
आस्की (ASCII())	हे फंक्शन अक्षराचे आस्की मूल्य शोधण्यासाठी वापरले जाते.
कॅर (CHAR())	आस्की कोडवर आधारित कॅरेक्टर मिळवते
कॅरइंडेक्स (CHARINDEX())	दिलेल्या स्ट्रिंगमधील सबस्ट्रिंगची स्थिती मिळवते.
कॉन्कॅट (CONCAT())	हे फंक्शन दोन शब्द किंवा स्ट्रिंग जोडण्यासाठी वापरले जाते
एस्केप (ESCAPE())	एस्केप कॅरेक्टरसह मजकूर मिळवते.
लेंथ (LEN())	दिलेल्या स्ट्रिंगची लांबी मिळवते.
लोवर (LOWER())	हे फंक्शन अप्पर केस स्ट्रिंगला लोअर केसमध्ये रूपांतरित करण्यासाठी वापरले जाते.
एलट्रिम (LTRIM())	हे स्ट्रिंग फंक्शन दिलेल्या मूळ स्ट्रिंगच्या डावीकडून दिलेला कॅरेक्टर किंवा स्ट्रिंग कट करते. हे निर्दिष्ट स्ट्रिंगच्या डावीकडील स्पेस देखील काढून टाकते.

फंक्शन (Function)	डिस्क्रिप्शन (Description)
एलपॅड (LPAD())	हे फंक्शन दिलेले चिन्ह जोडून दिलेल्या आकाराची स्ट्रिंग बनवण्यासाठी वापरले जाते.
मिड (MID())	हे स्ट्रिंग फंक्शन मूळ स्ट्रिंगच्या दिलेल्या स्थानावरून सब-स्ट्रिंग काढते.
एनकॅर (NCHAR())	नंबर कोडवर आधारित युनिकोड वर्ण मिळवते.
पोजिशन (POSITION())	हे स्ट्रिंग फंक्शन मुख्य स्ट्रिंगमध्ये दिलेल्या स्ट्रिंगच्या पहिल्या घटनेची स्थिती शोधते.
रिपीट (REPEAT())	हे स्ट्रिंग फंक्शन दिलेली स्ट्रिंग किंवा कॅरेक्टर दिलेल्या संख्येपर्यंत लिहिते.
रिप्लेस (REPLACE())	निर्दिष्ट स्ट्रिंगमधील सबस्ट्रिंगच्या सर्व घटना बदलून नवीन स्ट्रिंग मिळवते.
रिव्हर्स (REVERSE())	उलट स्ट्रिंग मिळवते.
राईट (RIGHT())	वास्तविक(वर्तमान) स्ट्रिंगमधून सर्वात उजवीकडे कॅरेक्टर मिळवते.
आरट्रिम (RTRIM())	सर्व अनुगामी रिक्त जागा काढून टाकल्यानंतर स्ट्रिंग मिळवते.
आरपॅड (RPAD())	हे स्ट्रिंग फंक्शन दिलेल्या स्ट्रिंगच्या उजवीकडे दिलेले चिन्ह जोडते.
स्पेस (SPACE())	हे स्ट्रिंग फंक्शन स्पेसची निर्दिष्ट संख्या जोडते.
सबएसटीआर (SUBSTR())	हे स्ट्रिंग फंक्शन मूळ स्ट्रिंगच्या दिलेल्या स्थानावरून सब-स्ट्रिंग काढते.
सबस्ट्रिंग (SUBSTRING())	हे फंक्शन नमूद केलेल्या आकार आणि दिलेल्या स्ट्रिंगमधून वर्णमाला शोधण्यासाठी वापरले जाते.
ट्रिम (TRIM())	हे फंक्शन स्ट्रिंगमधून दिलेले चिन्ह कापण्यासाठी वापरले जाते.
अप्पर (UPPER())	हे स्ट्रिंग फंक्शन वापरकर्त्यांना निर्दिष्ट स्ट्रिंगला अप्पर केस अक्षरांमध्ये रूपांतरित करण्यास अनुमती देते.

4. स्ट्रिंग पॅटर्न मॅचिंग ऑपरेटर (String Pattern Matching Operator)

लाईक ऑपरेटर(LIKE operator)

कॉलममध्ये असलेल्या विविध प्रकारचे पॅटर्न शोधण्यासाठी वेयर(WHERE) क्लॉजमध्ये लाईक (LIKE) ऑपरेटरचा वापर केला जातो.

लाईक (LIKE) ऑपरेटरच्या बरोबर दोन वाइल्डकार्ड वापरले जातात :

1. % (Percent) : शून्य, एक किंवा अनेक कॅरेक्टर सूचित करते.

वाक्यरचना(Syntax)

```
SELECT column1, column2, ...
FROM table_name
WHERE columnN LIKE pattern;
```

उदाहरण

- SELECT * FROM Customers
WHERE CustomerName LIKE 'a%'; --या उदाहरणात आपल्याला Customers टेबल मधील 'a' ने सुरू होणाऱ्या सर्व CustomerName आउटपुट म्हणून भेटणार आहेत.
- SELECT * FROM Customers
WHERE CustomerName LIKE '%a%'; --या उदाहरणात आपल्याला Customers टेबल मधील 'a' ने संपणारी सर्व CustomerName आउटपुट म्हणून भेटणार आहेत.
- SELECT * FROM Customers
WHERE CustomerName LIKE 'p%a%'; --या उदाहरणात आपल्याला Customers टेबल मधील 'p' ने सुरू होणारी आणि 'a' ने संपणारी आणि या दोन्ही कॅरेक्टर मध्ये शून्य, एक किंवा अनेक कॅरेक्टर असू शकतात अशी सर्व CustomerName आउटपुट म्हणून भेटणार आहेत.
- SELECT * FROM Customers
WHERE city LIKE '%a%'; --या उदाहरणात आपल्याला Customers टेबल मधील 'a' कॅरेक्टर असलेल्या सर्व city आउटपुट म्हणून भेटणार आहेत.

2. _ (Underscore) : एक कॅरेक्टर दर्शवते. ते कोणतेही कॅरेक्टर किंवा संख्या असू शकते, परंतु प्रत्येक _ एक आणि फक्त एक कॅरेक्टर दर्शवतो.

उदाहरण

```
SELECT * FROM Customers
WHERE city LIKE 'M_mb_';
```

--या उदाहरणात आपल्याला Customers टेबल मधील 'M' ने सुरू होणारी आणि त्यानंतर एक वाइल्डकार्ड कॅरेक्टर नंतर 'mb' आणि नंतर दोन वाइल्डकार्ड कॅरेक्टर असलेल्या सर्व city आउटपुट म्हणून भेटणार आहेत.

5. अॅग्रीगेट फंक्शन (Aggregate functions): एसक्यूएल विविध प्रकारच्या अॅग्रीगेट फंक्शनला समर्थन देते, ज्याचे खाली वर्णन केले आहे:

फंक्शन (Function)	डिस्क्रिप्शन (Description)
काउंट (COUNT())	डाटाबेस टेबलमधील पंक्तींची संख्या मोजण्यासाठी काउंट फंक्शन वापरले जाते.
सम (SUM())	सम फंक्शन कॉलम मधील व्हॅल्यूची एकूण बेरीज मिळवते.
अॅव्हरेज (AVG())	अॅव्हरेज फंक्शन न्यूमरिकल कॉलमच्या सरासरीची गणना करते.
मीन (MIN())	मीन फंक्शन निवडलेल्या कॉलम मधील सर्वात लहान व्हॅल्यू मिळवते.
मॅक्स (MAX())	मॅक्स फंक्शन निवडलेल्या कॉलम मधील सर्वात मोठे व्हॅल्यू मिळवते.

3.5.1 व्ह्यू (VIEW): एसक्यूएल मधील व्ह्यू हे व्हर्च्युअल टेबल म्हणून गणली जातात. एका व्ह्यू मध्ये रो आणि कॉलम देखील असतात. व्ह्यू तयार करण्यासाठी, आपण डाटाबेसमध्ये उपस्थित असलेल्या एक किंवा

अधिक टेबलमधून फील्ड निवडू शकतो. व्ह्यूमध्ये विशिष्ट स्थितीवर आधारित विशिष्ट रो असू शकतात किंवा टेबलच्या सर्व रो असू शकतात.

1. क्रिएटिंग व्ह्यू (CREATE VIEW): CREATE VIEW (क्रिएट व्ह्यू) स्टेटमेंट वापरून व्ह्यू तयार केले जाऊ शकते. आपण एकाच टेबल किंवा अनेक टेबल्समधून व्ह्यू तयार करू शकतो.

वाक्यरचना (Syntax)

```
CREATE VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE condition;
```

उदाहरण

```
CREATE VIEW ViewDetail AS
SELECT NAME, ADDRESS
FROM StudentDetail
WHERE SID < 10;
```

2. अपडेट व्ह्यू (UPDATE VIEW): जर तुम्हाला व्ह्यूमध्ये विद्यमान डाटा अपडेट करायचा असेल, तर अपडेट व्ह्यू स्टेटमेंट वापरा.

वाक्यरचना (Syntax)

```
UPDATE view_name
SET column1 = value1, column2 = value2....., columnN = valueN
WHERE [condition];
```

Or

```
CREATE OR REPLACE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

उदाहरण

```
CREATE OR REPLACE VIEW MarksView AS
SELECT StudentD.NAME, StudentD.ADDRESS, StudentM.MARKS, StudentM.AGE
FROM StudentD, StudentM
WHERE StudentD.NAME = StudentM.NAME;
```

3. ड्रॉप व्ह्यू (DROP VIEW): ड्रॉप व्ह्यू स्टेटमेंट वापरून व्ह्यू हटवले जाऊ शकते.

वाक्यरचना (Syntax)

```
DROP VIEW view_name;
```

उदाहरण

```
DROP VIEW StudentDetail;
```

3.5.2 सिक्वेन्स (SEQUENCE): एसक्यूएल मधील सिक्वेन्स हे डाटाबेस ऑब्जेक्ट्स आहेत जे युनिक इंटीजर व्हॅल्यूची सिक्वेन्स तयार करते. सिक्वेन्स हे डाटाबेसमध्ये वारंवार वापरले जातात कारण अनेक ॲप्लिकेशन्सना आवश्यक असते की टेबलमधील प्रत्येक पंक्तीमध्ये युनिक व्हॅल्यू असणे आवश्यक आहे आणि सिक्वेन्स त्यांना युनिक व्हॅल्यू जनरेट करण्याचा सोपा मार्ग प्रदान करते. न्यूमेरिकल व्हॅल्यूचा क्रम परिभाषित अंतराने चढत्या किंवा उतरत्या क्रमाने जनरेट केला जातो आणि जेव्हा तो मॅक्स व्हॅल्यू ओलांडतो तेव्हा रीस्टार्ट करण्यासाठी कॉन्फिगर केला जाऊ शकतो.

1. क्रिएटिंग सिक्वेन्स (Creating Sequences): CREATE SEQUENCE (क्रिएट सिक्वेन्स) स्टेटमेंट वापरून सिक्वेन्स तयार केले जाऊ शकते.

वाक्यरचना (Syntax)

```
CREATE SEQUENCE sequence_name
START WITH initial_value
INCREMENT BY increment_value
MINVALUE minimum_value
MAXVALUE maximum_value
CYCLE|NOCYCLE
CACHE cache_value|NOCACHE;
```

Initial_value: इनिशियल व्हॅल्यू(initial_value) हे सिक्वेन्सचे प्रारंभिक व्हॅल्यू दर्शवते. या व्हॅल्यूसह सिक्वेन्स ऑब्जेक्ट आरंभ केला जातो आणि या मूल्यावर पुढील वाढ किंवा घट केली जाते.

Increment_value: इन्क्रिमेंट व्हॅल्यू (increment_value) हे व्हॅल्यू किंवा इंडेक्स दर्शवते ज्याद्वारे सिक्वेन्स ऑब्जेक्टचे पुढील व्हॅल्यू प्राप्त केल्यावर प्रत्येक वेळी सिक्वेन्सचे प्रारंभिक व्हॅल्यू वाढले जाईल.

Minimum_value: मिनिमम व्हॅल्यू (minimum_value) हे मिनिमम व्हॅल्यू दर्शवते जे सिक्वेन्स ऑब्जेक्ट पर्यंत जाऊ शकते. हे लोअर बाउंड म्हणून कार्य करते.

Maximum_value: मॅक्सिमम व्हॅल्यू (Maximum_value) हे सिक्वेन्स ऑब्जेक्टची मॅक्सिमम व्हॅल्यू दर्शवते. हे वरच्या बाउंड म्हणून कार्य करते.

Cycle/Nocycle: नोसायकल ही डीफॉल्ट आहे. जर सायकल असेल तर कमाल/किमान मूल्य गाठल्यानंतरही सिक्वेन्स व्हॅल्यू निर्माण करेल.

CACHE: कॅशे पर्याय सिक्वेन्स व्हॅल्यूना जलद प्रवेश देण्यासाठी मेमरीमधील सिक्वेन्स व्हॅल्यू कॅश करतो

उदाहरण

```
CREATE SEQUENCE seq2
start with 100
increment by 1
min value 1
max value 800
cycle;
```

2. अल्टर सिक्वेन्स (ALTER SEQUENCE): नवीन मॅक्सिमम(Maximum) व्हॅल्यू सेट करण्यासाठी आपण सिक्वेन्स बदलू शकतो. आपण अल्टर सिक्वेन्स कमांडने स्टार्ट व्हॅल्यू बदलू शकत नाही. स्टार्ट व्हॅल्यू बदलण्यासाठी आपल्याला सिक्वेन्स ड्रॉप करून पुन्हा तयार करणे आवश्यक आहे.

वाक्यरचना (Syntax)

```
ALTER SEQUENCE sequencename
INCREMENT BY increment_value
MINVALUE maximum_value
MINVALUE minimum_value
CYCLE|NOCYCLE
CACHE cache_value|NOCACHE ;
```

उदाहरण

```
ALTER SEQUENCE seq2
INCREMENT BY 3
MINVALUE 3
```

MINVALUE 400

CYCLE

CACHE 5;

3. ड्रॉप सिक्वेन्स (DROP SEQUENCE): ड्रॉप सिक्वेन्स स्टेटमेंट आपल्याला डाटाबेसमधून सिक्वेन्स काढण्याची परवानगी देते.

वाक्यरचना(Syntax)

DROP SEQUENCE Sequencename;

उदाहरण

DROP SEQUENCE seq2;

3.5.3. इंडेक्स (INDEX): इंडेक्स हा एक स्कीमा ऑब्जेक्ट आहे जलद डाटा ऍक्सेस करण्यासाठी किंवा पॉइंटर वापरून रो रिट्रिव्हल करण्यासाठी वेगवान पर्याय प्राप्त करतो. इंडेक्स डाटाबेसद्वारे ऑटोमॅटिकली सेव्ह केले जातात. ते रॅपिड पाथ ऍक्सेस पद्धती वापरून डिस्क इनपुट/आउटपुट कमी करते .

टाइप ऑफ इंडेक्स (TYPE OF INDEX)

एसक्यूएल मध्ये इंडेक्सचे दोन प्रकार आहेत.

I. क्लस्टर इंडेक्स(Clustered Index): क्लस्टर्ड इंडेक्स हा इंडेक्सचा प्रकार आहे जो रोचा फिजिकल सॉर्टिंग ऑर्डर लावतो. एका टेबलमध्ये एकच क्लस्टर इंडेक्स असू शकते जे टेबलला युनिक व्हॅल्यू स्टोअर करण्यासाठी मदत करते.

II. नॉन क्लस्टर इंडेक्स(Non-Clustered Index): नॉन-क्लस्टर्ड इंडेक्स ही एक किंवा अधिक निवडलेल्या कॉलमची पुनर्क्रमण करणारी टेबलमध्ये संग्रहित केलेल्या डाटापासून वेगळी अनुक्रमणिका संरचना आहे. नॉन-क्लस्टर्ड इंडेक्स क्लस्टर केलेल्या निर्देशांकात समाविष्ट नसलेल्या वारंवार वापरल्या जाणाऱ्या क्हेरींचे कार्यप्रदर्शन सुधारण्यासाठी तयार केले जातात.

1. क्रिएटिंग इंडेक्स (CREATE INDEX)

CREATE INDEX (क्रिएट इंडेक्स) स्टेटमेंट वापरून इंडेक्स तयार केले जाऊ शकते.

वाक्यरचना (Syntax)

CREATE INDEX <index_name>

ON <table_name> (column1, column2, ...);

उदाहरण

CREATE INDEX myindex

ON Student (Rollno);

2. ड्रॉप इंडेक्स (DROP INDEX): ड्रॉप इंडेक्स स्टेटमेंट वापरून इंडेक्स हटवले जाऊ शकते.

वाक्यरचना (Syntax)

DROP indexname;

उदाहरण

DROP INDEX myindex;

References:

1. <https://nptel.ac.in/courses/106105175>
2. <https://www.w3schools.com/sql/>
3. <https://www.tutorialspoint.com/sql/index.htm>
4. Database System Concepts by Henry F. Korth McGraw Hill Education

युनिट - 4

PL / SQL प्रोग्रामिंग (PL/SQL Programming)

विषय निष्पत्ती (Course Outcome): दिलेल्या ॲप्लिकेशनसाठी PL/SQL कोड लागू करा.

घटक निष्पत्ती (Theory Learning Outcome):

1. PL-SQL मध्ये कंट्रोल स्ट्रक्चर्स वापरा.
2. विविध प्रकारचे अपवाद हाताळा.
3. विविध प्रकारचे कर्सर समजावून सांगा.
4. दिलेल्या समस्येवर प्रक्रिया, कार्य तयार करा.
5. ट्रिगरचे प्रकार उदाहरणांसह स्पष्ट करा

4.0 परिचय

- PL/SQL ही डाटाबेस प्रोग्रामिंग भाषा आहे. PL/SQL म्हणजे SQL च्या प्रक्रियात्मक भाषा विस्तार. PL/SQL हे प्रोग्रामिंग भाषांच्या प्रक्रियात्मक वैशिष्ट्यांसह SQL चे संयोजन आहे.
- SQL च्या क्षमता वाढवण्यासाठी PL/SQL हे 90 च्या दशकाच्या सुरुवातीला Oracle कॉर्पोरेशनने विकसित केले होते. PL/SQL स्टेटमेंटवर प्रक्रिया करण्यासाठी ओरॅकल PL/SQL इंजिन वापरते. PL/SQL कोड क्लायंट सिस्टममध्ये किंवा सर्व्हरच्या बाजूला संग्रहित केला जाऊ शकतो.
- PL/SQL ला कमांड लाइन SQL* Plus इंटरफेस वरून देखील थेट कॉल केले जाऊ शकते.
- PL/SQL ही एक प्रक्रियात्मक भाषा आहे जी विशेषतः त्याच्या वाक्यरचनामध्ये SQL विधाने स्वीकारण्यासाठी डिझाइन केलेली आहे. PL/SQL प्रोग्राम युनिट्स ओरॅकल डाटाबेस सर्व्हरद्वारे संकलित केली जातात आणि डाटाबेसमध्ये संग्रहित केली जातात. आणि रन-टाइममध्ये, PL/SQL आणि SQL दोन्ही एकाच सर्व्हर प्रक्रियेत चालतात, इष्टतम कार्यक्षमता आणतात. PL/SQL ला Oracle डाटाबेसची मजबूती, सुरक्षितता आणि पोर्टेबिलिटी आपोआप वारसा मिळते.

4.1 PL/SQL ची ओळख

PL/SQL हे Oracle मधील प्रमुख तंत्रज्ञानांपैकी एक आहे. PL/SQL स्ट्रक्चर्ड केरी लॅंग्वेजच्या रिलेशनल डाटा एक्सेसेसना लवचिक एम्बेड केलेल्या प्रक्रियात्मक स्थितीसह क्षमता आहे. आणि ते डाटाबेस इंजिनमध्येच जटिल केरी आणि प्रोग्रामॅटिक लॉजिक चालवते. हे डाटाबेस-चालित अनुप्रयोगांची चपळता, कार्यक्षमता आणि कार्यप्रदर्शन वाढवते.

PL/SQL म्हणजे प्रक्रियात्मक भाषा/संरचित केरी भाषा. PL/SQL प्रक्रियात्मक आदेशांचा एक संच (IF स्टेटमेंट्स, लूप, असाइनमेंट) ऑफर करते, ब्लॉक्समध्ये आयोजित (खाली स्पष्ट केले आहे), जे SQL ची पोहोच वाढवतात. PL/SQL हे ओरॅकल डाटाबेससह काम करणाऱ्या विकासकांसाठी डिझाइन केलेले एसक्यूएलचे ओरॅकलचे विस्तार आहे.

4.1.1 PL/SQL चे फायदे

PL/SQL चे विविध फायदे खाली दिले आहेत:

- ब्लॉक स्ट्रक्चर्स: PL/SQL मध्ये कोडचे ब्लॉक्स असतात, जे एकमेकांमध्ये नेस्ट केले जाऊ शकतात. प्रत्येक ब्लॉक टास्कचे युनिट किंवा लॉजिकल मॉड्यूल बनवतो. PL/SQL ब्लॉक्स डाटाबेसमध्ये संग्रहित केले जाऊ शकतात आणि पुन्हा वापरले जाऊ शकतात.
- प्रक्रियात्मक भाषा क्षमता: PL/SQL मध्ये प्रक्रियात्मक भाषेची रचना असते जसे की सशर्त विधाने (अन्यतर विधाने) आणि लूप जसे की (FOR लूप).

- उत्तम कार्यप्रदर्शन: PL/SQL इंजिन एकाच वेळी एकाच ब्लॉकच्या रूपात एकाधिक SQL स्टेटमेंट्सवर प्रक्रिया करते, ज्यामुळे नेटवर्क रहदारी कमी होते आणि अनुप्रयोगासाठी उच्च कार्यक्षमता प्रदान करते.
- त्रुटी हाताळणे: PL/SQL प्रोग्रामच्या अंमलबजावणीदरम्यान त्रुटी किंवा अपवाद प्रभावीपणे हाताळते. एकदा अपवाद पकडला गेला की, अपवादाच्या प्रकारावर अवलंबून विशिष्ट क्रिया केल्या जाऊ शकतात किंवा वापरकर्त्याला संदेशासह ते प्रदर्शित केले जाऊ शकतात.

4.1.2 PL/SQL ब्लॉक स्ट्रक्चर

PL/SQL ही ब्लॉक-संरचित भाषा आहे. प्रत्येक PL/SQL प्रोग्राममध्ये SQL आणि PL/SQL स्टेटमेंट असतात. PL/SQL मधील मूलभूत युनिट एक ब्लॉक आहे. तार्किकदृष्ट्या संबंधित विधानांचा संच ब्लॉक बनवतो.

PL/SQL ब्लॉकमध्ये चित्र 4.1 मध्ये दाखवल्याप्रमाणे तीन विभाग असतात.

1. **घोषणा विभाग:** घोषणा विभाग DECLARE या कीवर्डने सुरू होतो. हा विभाग पर्यायी आहे आणि व्हेरिएबल्स, स्थिरांक, रेकॉर्ड आणि कर्सर घोषित करण्यासाठी वापरला जातो
2. **एक्झिक्युशन सेक्शन:** एक्झिक्युशन सेक्शन BEGIN या कीवर्डने सुरू होतो आणि END ने समाप्त होतो. हा एक अनिवार्य विभाग आहे. प्रोग्राम लॉजिक येथे लिहिले आहे. लूप, कंडिशनल स्टेटमेंट आणि एसक्यूएल स्टेटमेंट्स सारख्या रचना येथे लिहिलेल्या आहेत.
3. **अपवाद विभाग:** अपवाद विभाग EXCEPTION या कीवर्डने सुरू होतो आणि तो ऐच्छिक आहे. प्रोग्राममधील कोणत्याही त्रुटी या विभागात हाताळल्या जातात.

वरील तीन विभागातील प्रत्येक विधान अर्धविराम (;) ने समाप्त होणे आवश्यक आहे.

```
DECLARE
/* घोषणात्मक विभाग: चल, प्रकार आणि स्थानिक उपप्रोग्राम्स.*/
BEGIN
/* एक्झिक्युटेबल विभाग: प्रक्रियात्मक आणि SQL स्टेटमेंट्स येथे जातात.*/
/* ब्लॉकचा हा एकमेव विभाग आवश्यक आहे.*/
EXCEPTION
/*Exception handling section: error handling statements go here.*/
```

आकृती 4.1 : PL/SQL ब्लॉक स्ट्रक्चर

पहिले PL/SQL उदाहरण 'हॅलो वर्ल्ड':

```
DECLARE
message varchar2(20):= 'Hello, World!';
BEGIN
dbms_output.put_line(message);
END;
/
```

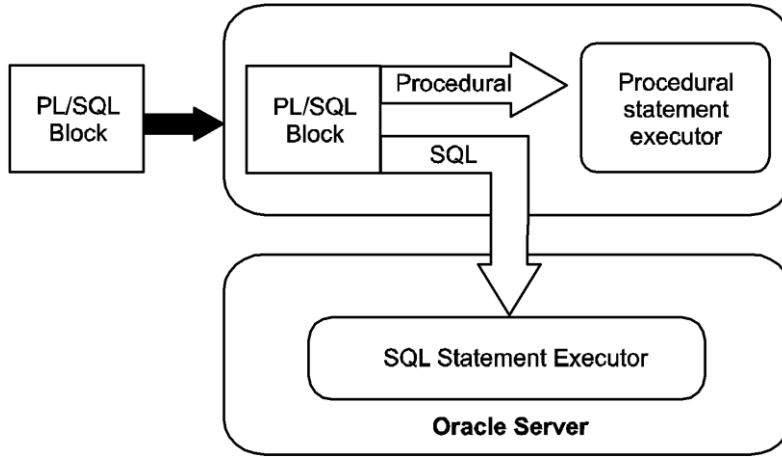
शेवट; ओळ PL/SQL ब्लॉकच्या शेवटी सिग्नल करते. एसक्यूएल कमांड लाइनवरून कोड चालवण्यासाठी, तुम्हाला कोडच्या शेवटच्या ओळीनंतर पहिल्या रिकाम्या ओळीच्या सुरुवातीला / टाईप करावे लागेल. जेव्हा वरील कोड SQL प्रॉम्प्टवर कार्यान्वित केला जातो तेव्हा ते खालील आउटपुट तयार करते:

Hello World!

PL/SQL procedure successfully completed.

PL/SQL अंमलबजावणी

- PL/SQL संकलन आणि रन-टाइम सिस्टम हे एक इंजिन आहे जे PL/SQL ब्लॉक्स आणि सबप्रोग्राम्स संकलित आणि कार्यान्वित करते. इंजिन ओरॅकल सर्व्हरमध्ये किंवा ओरॅकल फॉर्म्स सारख्या ऍप्लिकेशन डेव्हलपमेंट टूलमध्ये स्थापित केले जाऊ शकते.
- आकृती. 4.2 PL/SQL इंजिन निनावी ब्लॉकवर प्रक्रिया करत असल्याचे दाखवते. PL/SQL इंजिन प्रक्रियात्मक विधाने कार्यान्वित करते परंतु ओरॅकल डाटाबेसमधील SQL इंजिनला SQL स्टेटमेंट पाठवते.
- PL/SQL हे एक तंत्रज्ञान आहे जे ओरॅकल सर्व्हरद्वारे आणि विशिष्ट Oracle साधनांद्वारे वापरले जाते. PL/SQL चे ब्लॉक्स पास केले जातात आणि PL/SQL इंजिनद्वारे त्यावर प्रक्रिया केली जाते, जे टूलमध्ये किंवा ओरॅकल सर्व्हरमध्ये असू शकतात.
- वापरले जाणारे इंजिन कोडचा PL/SQL ब्लॉक कोटून मागवला जातो त्या स्त्रोतावर अवलंबून असते.
- जेव्हा तुम्ही प्रो* प्रोग्राम, यूजर-एक्झिट, SQL*PLUS, किंवा सर्व्हर मॅनेजरमधून PL/SQL ब्लॉक सबमिट करता, तेव्हा Oracle सर्व्हरमधील PL/SQL इंजिन त्यावर प्रक्रिया करते.



आकृती. 4.2

- ते एसक्यूएल स्टेटमेंट वेगळे करते आणि ते एसक्यूएल स्टेटमेंट एक्झिक्यूटरकडे वैयक्तिकरित्या पाठवते.
- SQL स्टेटमेंट फिल्टर करून, ऍप्लिकेशनमधून ओरॅकल सर्व्हरवर फक्त एकच हस्तांतरण आवश्यक आहे.
- हे नेटवर्क रहदारी कमी करते आणि कार्यप्रदर्शन मोठ्या प्रमाणात वाढवते, विशेषतः क्लायंट/सर्व्हर नेटवर्कमध्ये.
- ओरॅकल टूलसमधील PL/SQL इंजिन ऍप्लिकेशन साइटवर PL/SQL ब्लॉकमधील केवळ प्रक्रियात्मक विधाने कार्यान्वित करते. हे या उद्देशासाठी स्थानिक PL/SQL इंजिनमधील प्रक्रियात्मक विधान एक्झिक्यूटर वापरते.

4.1.3 PL/SQL डाटा प्रकार

PL/SQL व्हेरिएबल्स, स्थिरांक आणि पॅरामीटर्समध्ये वैध डाटा प्रकार असणे आवश्यक आहे जे स्टोरेज फॉर्मॅट, मर्यादा आणि मूल्यांची वैध श्रेणी निर्दिष्ट करते.

1. PL/SQL मूलभूत डाटा प्रकार:

डाटा प्रकार	वर्णन (Description)	स्टोरेज (कमाल)
CHAR(size)	हा डाटा प्रकार निश्चित लांबीच्या वर्ण स्ट्रिंग मूल्य संचयित करण्यासाठी वापरला जातो. त्यात अक्षरे, संख्या आणि विशेष वर्ण असू शकतात.	255 Character
VARCHAR(size)/ VARCHAR2(size)	हा डाटा प्रकार व्हेरिएबल लांबी अल्फान्यूमेरिक डाटा संचयित करण्यासाठी वापरला जातो. जर CHAR डाटा प्रकार असेल तर ते अधिक लवचिक आहे. त्यात अक्षरे, संख्या आणि विशेष वर्ण असू शकतात.	4000 Character
Date()	हा डाटा प्रकार डेट आणि वेळ दर्शवण्यासाठी वापरला जातो. मानक स्वरूप DD-MON-YY आहे.	
Number (P,S)	हा डाटा प्रकार क्रमांक संचयित करण्यासाठी वापरला जातो (निश्चित किंवा फ्लोटिंग पॉइंट). P म्हणजे अचूकता, अंकाची कमाल लांबी. S म्हणजे स्केल, डीफॉल्टनुसार दशांश स्थानांची संख्या 0.	38 Digit Precision
LONG	हा डाटा प्रकार व्हेरिएबल लांबी कॅरेक्टर स्ट्रिंगमध्ये 2GB पर्यंत साठवण्यासाठी वापरला जातो.	2GB
RAW	हा डाटा प्रकार बायनरी डाटा जसे की डिजिटाइज्ड, पिक्चर किंवा इमेज साठवण्यासाठी वापरला जातो.	255 Bytes
LONG RAW	हा डाटा प्रकार बायनरी डाटा जसे की डिजिटाइज्ड, पिक्चर किंवा इमेज साठवण्यासाठी वापरला जातो	2GB

2. आगाऊ डाटा प्रकार:

डाटा प्रकार	वर्णन (Description)	स्टोरेज (कमाल)
BLOB	हा डाटा प्रकार बायनरी वर्ण जसे की गाणी, व्हिडिओ, प्रतिमा, तयार केलेला डाटा प्रकार संग्रहित करण्यासाठी आणि 4 GB पर्यंत ठेवण्यासाठी वापरला जातो.	4 GB
CLOB	हा डाटा प्रकार कॅरेक्टर व्हेरिएबल संचयित करण्यासाठी आणि 4 GB पर्यंत ठेवण्यासाठी वापरला जातो.	4 GB
BFile	हा डाटा प्रकार पॉइंटर व्हेरिएबल, पॉइंटरचा प्रकार, बाह्य फाइलचा संदर्भ यामधील मूल्य साठवण्यासाठी वापरला जातो.	4 GB
%Type	हा डाटा प्रकार टेबलमधील मूल्य अज्ञात डाटा प्रकार स्तंभ संग्रहित करण्यासाठी वापरला जातो, स्तंभ %type डाटा प्रकाराद्वारे ओळखला जातो. उदाहरणार्थ, emp.eno%type emp नाव टेबल आहे, eno हा अज्ञात डाटा प्रकार स्तंभ आहे आणि	

डाटा प्रकार	वर्णन (Description)	स्टोरेज (कमाल)
	मूल्य धारण करण्यासाठी % प्रकार हा डाटा प्रकार आहे.	
%RowType	हा डाटा प्रकार सारणीमधील सर्व स्तंभ मूल्य अज्ञात डाटा प्रकार संग्रहित करण्यासाठी वापरला जातो. सर्व स्तंभ %RowType डाटाटाइपद्वारे ओळखले जातात. उदाहरणार्थ, emp% rowtype emp नाव टेबल आहे, सर्व स्तंभ प्रकार % rowtype आहे.	
%RowID	RowID हा डाटा प्रकार आहे. RowID दोन प्रकारचे विस्तारित किंवा प्रतिबंधित आहे. विस्तारित परतावा 0 आणि प्रतिबंधित परतावा 1 अन्यथा पंक्ती क्रमांक परत करा	

4.1.4 व्हेरिएबल

- व्हेरिएबल म्हणजे दुसरे काहीही नसून स्टोरेज क्षेत्राला दिलेले नाव आहे जे आमचे प्रोग्राम हाताळू शकतात. माहिती PL/SQL प्रोग्राम आणि डाटाबेस दरम्यान व्हेरिएबल्सद्वारे प्रसारित केली जाते.
- PL/SQL मधील प्रत्येक व्हेरिएबलमध्ये विशिष्ट डाटा प्रकार असतो, जो व्हेरिएबलच्या मेमरीचा आकार आणि मांडणी ठरवतो; मूल्यांची श्रेणी जी त्या मेमरीमध्ये संग्रहित केली जाऊ शकते आणि ऑपरेशन्सचा संच जो व्हेरिएबलवर लागू केला जाऊ शकतो.
- PL/SQL व्हेरिएबलच्या नावामध्ये वैकल्पिकरित्या एक अक्षर असते ज्यानंतर अधिक अक्षरे, अंक, डॉलर चिन्हे, अंडरस्कोअर आणि संख्या चिन्हे असतात आणि ती 30 वर्णापेक्षा जास्त नसावीत. डीफॉल्टनुसार, व्हेरिएबलची नावे केस-संवेदी नसतात. तुम्ही व्हेरिएबल नाव म्हणून आरक्षित PL/SQL कीवर्ड वापरू शकत नाही.
- प्रत्येक व्हेरिएबलचा त्याच्याशी संबंधित विशिष्ट डाटा प्रकार असतो. तो डाटा प्रकार असू शकतो:
 - डाटाबेस स्तंभांसाठी SQL द्वारे वापरल्या जाणाऱ्या प्रकारांपैकी एक.
 - PL/SQL मध्ये वापरलेला सामान्य प्रकार जसे की NUMBER.
 - %TYPE सारख्या काही डाटाबेस स्तंभाच्या प्रकाराप्रमाणेच असल्याचे घोषित केले.
- सर्वात सामान्यतः वापरला जाणारा सामान्य प्रकार NUMBER आहे. NUMBER प्रकारातील व्हेरिएबल्समध्ये पूर्णांक किंवा वास्तविक संख्या असू शकते. सर्वात सामान्यपणे वापरलेला वर्ण स्ट्रिंग प्रकार VARCHAR(n) आहे, जेथे n ही बाइट्समधील स्ट्रिंगची कमाल लांबी आहे. ही लांबी आवश्यक आहे, आणि कोणतेही डीफॉल्ट नाही.
- उदाहरणार्थ

```
DECLARE
    sal NUMBER;
    LastName VARCHAR(20);
```

टीप: PL/SQL BOOLEAN व्हेरिएबल्सना परवानगी देते, जरी Oracle समर्थन देत नसले तरी. डाटाबेस स्तंभांसाठी एक प्रकार म्हणून BOOLEAN.

- बऱ्याच प्रकरणांमध्ये, PL/SQL व्हेरिएबलचा वापर विद्यमान संबंधात संग्रहित डाटा हाताळण्यासाठी केला जाईल. या प्रकरणात, व्हेरिएबलमध्ये %TYPE ऑपरेटर वापरून रिलेशन कॉलम सारखाच डाटा प्रकार असणे आवश्यक आहे.

- उदाहरणार्थ

```
DECLARE
MyLastName EMP.LastName%TYPE;
```

- PL/SQL व्हेरिएबल MyLastName जो काही प्रकार EMP संबंधात LastName स्तंभासाठी घोषित केला गेला होता. व्हेरिएबलमध्ये एक प्रकार देखील असू शकतो जो अनेक फील्डसह रेकॉर्ड आहे. असे व्हेरिएबल घोषित करण्याचा सर्वात सोपा मार्ग म्हणजे संबंध नावावर %ROWTYPE वापरणे.
- परिणाम हा एक रेकॉर्ड प्रकार आहे ज्यामध्ये फील्डची नावे आणि संबंधांच्या गुणधर्माप्रमाणेच प्रकार आहेत
- उदाहरणार्थ

```
DECLARE
Dept_Info DEPT%ROWTYPE;
```

Dept_Info व्हेरिएबलला DeptNo, DeptName आणि Loc फील्डसह रेकॉर्ड बनवते, असे गृहीत धरून की संबंधात स्कीमा DEPT(DeptNo, DeptName आणि Loc) आहे.

व्हेरिएबल घोषित करण्यासाठी सामान्य वाक्यरचना :

```
variable_name datatype [NOT NULL := value ];
```

उदाहरणार्थ

```
DECLARE
a NUMBER := 3;
BEGIN
a := a + 1;
END;
run;
```

- हा प्रोग्राम चालवताना कोणताही प्रभाव पडत नाही, कारण डाटाबेसमध्ये कोणतेही बदल नाहीत
- PL/SQL मध्ये व्हेरिएबल स्कोप:
- PL/SQL ब्लॉक्सच्या नेस्टिंगला अनुमती देते म्हणजेच, प्रत्येक प्रोग्राम ब्लॉकमध्ये दुसरा अंतर्गत ब्लॉक असू शकतो. जर व्हेरिएबल आतील ब्लॉकमध्ये घोषित केले असेल, तर ते बाह्य ब्लॉकमध्ये प्रवेश करण्यायोग्य नाही. तथापि, जर व्हेरिएबल घोषित केले असेल आणि बाह्य ब्लॉकमध्ये प्रवेशयोग्य असेल, तर ते सर्व नेस्टेड अंतर्गत ब्लॉक्ससाठी देखील प्रवेशयोग्य आहे.
- व्हेरिएबल स्कोपचे दोन प्रकार आहेत

लोकल व्हेरिएबल्स: व्हेरिएबल्स आतील ब्लॉकमध्ये घोषित केले जातात आणि बाह्य ब्लॉक्समध्ये प्रवेशयोग्य नाहीत.

ग्लोबल व्हेरिएबल्स: सर्वात बाहेरील ब्लॉक किंवा पॅकेजमध्ये घोषित व्हेरिएबल्स.

- खालील उदाहरण स्थानिक आणि ग्लोबल व्हेरिएबल्सचा वापर त्याच्या सोप्या स्वरूपात दाखवते

```
DECLARE
-- Global variables
num1 number := 95;
num2 number := 85;
BEGIN
dbms_output.put_line('Outer Variable num1: ' || num1);
dbms_output.put_line('Outer Variable num2: ' || num2);
DECLARE
-- Local variables
```

```

        num1 number := 195;
        num2 number := 185;
    BEGIN
    dbms_output.put_line('Inner Variable num1: ' || num1);
    dbms_output.put_line('Inner Variable num2: ' || num2);
    END;
END;
/

```

जेव्हा वरील कोड कार्यान्वित केला जातो तेव्हा तो पुढील परिणाम देतो

```

Outer Variable num1: 95
Outer Variable num2: 85
Inner Variable num1: 195
Inner Variable num2: 185
PL/SQL procedure successfully completed.

```

नोटपॅडमध्ये दोन नंबर जोडण्याचा कार्यक्रम.

```

DECLARE
    A NUMBER:=5;
    B NUMBER:=5;
BEGIN
    A:=A+B;
END;

```

टाईप केल्यानंतर जर तुम्हाला बदल करायचे असतील तर तुम्ही SQL च्या edit कमांडचा वापर करून संपादक उघडू शकता.

```
SQL>edit
```

वरील कोड प्रकार कार्यान्वित करण्यासाठी

```

SQL>.
SQL>run;
Or
SQL>/

```

वरील प्रोग्राम कोणतेही आउटपुट प्रिंट करत नाही कारण कोणतेही प्रिंट स्टेटमेंट लिहिलेले नाही.

स्क्रीनवर संदेश कसे प्रदर्शित करावे? (How to display message on screen)

- **DBMS_OUTPUT:** एक पॅकेज आहे ज्यामध्ये अनेक प्रक्रिया आणि कार्ये समाविष्ट आहेत जी बफरमध्ये माहिती जमा करतात जेणेकरून ती नंतर पुनर्प्राप्त केली जाऊ शकते. ही कार्ये वापरकर्त्याला संदेश प्रदर्शित करण्यासाठी देखील वापरली जाऊ शकतात.
- **PUT_LINE:** पॅकेज बफरमध्ये माहितीचा एक तुकडा ठेवा आणि त्यानंतर शेवटच्या ओळीचा मार्कर ठेवा. हे वापरकर्त्याला संदेश प्रदर्शित करण्यासाठी देखील वापरले जाऊ शकते. पुट_लाइन कॅरेक्टर डाटा प्रकाराच्या एका पॅरामीटरची अपेक्षा करते. संदेश प्रदर्शित करण्यासाठी वापरल्यास, तो संदेश 'स्ट्रिंग' आहे.

उदाहरणार्थ

वापरकर्त्याला संदेश प्रदर्शित करण्यासाठी SERVEROUTPUT चालू वर सेट केले पाहिजे. SERVEROUTPUT हे sql*plus पर्यावरण पॅरामीटर आहे जे PUT_LINE फंक्शनला पॅरामीटर म्हणून दिलेली माहिती प्रदर्शित करते.

उदाहरणार्थ

SET SERVEROUTPUT ON

उदाहरणार्थ

नोटपॅडमध्ये दोन नंबर जोडण्यासाठी खालील कोड टाका.

```

DECLARE
    A NUMBER:=5;
    B NUMBER:=5;
BEGIN
    A:=A+B;
    Dbms_output.put_line('The Addition of two numbers is:-' || A);
END;
```

टाईप केल्यानंतर जर तुम्हाला बदल करायचे असतील तर तुम्ही SQL च्या edit कमांडचा वापर करून संपादक उघडू शकता.

SQL>edit

वरील कोड प्रकार कार्यान्वित करण्यासाठी:

SQL>.

SQL>run;

Or

SQL>/

The Addition of two numbers is :- 10

PL/SQL procedure successfully completed.

वापरकर्त्याकडून इनपुट स्वीकारा: (How to accept input from user)

- वापरकर्त्याकडून इनपुट स्वीकारण्यासाठी तुम्ही व्हेरिएबलच्या आधी '&' ऑपरेटर वापरू शकता. SQL प्रॉम्प्टवर क्लिक करून रन करण्यासाठी "व्हेरिएबल नेम:=&व्हेरिएबल नेम;" वापरा.
- Oracle 11g Express Edition साठी तुम्ही "variable name :=variable name;" लिहू शकता. इनपुट स्वीकारल्याबद्दल.
- उदाहरणार्थ नोटपॅडमध्ये दोन नंबर जोडण्यासाठी खालील कोड टाका.

```

DECLARE
    A NUMBER;
    B NUMBER;
BEGIN
    A:=&a;
    B:=&b;
    A:=A+B;
    Dbms_output.put_line ('The Addition of two numbers is:-' || A);
END;
```

टाईप केल्यानंतर जर तुम्हाला बदल करायचे असतील तर तुम्ही SQL च्या edit कमांडचा वापर करून संपादक उघडू शकता.

SQL>edit

वरील कोड प्रकार कार्यान्वित करण्यासाठी:

SQL>.

SQL>run;

Or

```
SQL>/
Enter value for A:6
Old 5: A:=&a;
New 5 : A:=6
Enter value for B:7
Old 6: B:=&b
New 6 : B:=7
The Addition of two numbers is :- 13
PL/SQL procedure successfully completed.
```

किंवा

The screenshot shows the Oracle Application Express interface. The top navigation bar includes 'Home', 'Application Builder', 'SQL Workshop', 'Team Development', and 'Administration'. The 'SQL Workshop' section is active, showing 'SQL Commands'. Below the navigation bar, there are buttons for 'Autocommit', 'Rows' (set to 10), 'Save', and 'Run'. The main area contains a PL/SQL block:

```
DECLARE
  A NUMBER;
  B NUMBER;
BEGIN
  A:=:a;
  B:=:b;
  A:=A+B;
  DBMS_OUTPUT.PUT_LINE ('The Addition of two numbers is:-' || A);
END;
```

Below the code editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the output: 'The Addition of two numbers is:-3' and 'Statement processed.'

PL/SQL मध्ये सिलेक्ट स्टेटमेंट वापरणे:

PL/SQL मधील सिलेक्ट स्टेटमेंट वापरले जाऊ शकते जर केरी एकच पंक्ती दर्शवते. जर केरीने अनेक पंक्ती दिल्या तर तुम्हाला कर्सर वापरावा लागेल.

वाक्यरचना :

```
Select attribute names into variable names
From table name
Where condition;
```

SELECT क्लॉज नंतर, आम्ही INTO क्लॉज व्हेरिएबल नावे वापरणे आवश्यक आहे, SELECT क्लॉजमधील प्रत्येक विशेषतासाठी एक, ज्यामध्ये पुनर्प्राप्त केलेल्या ट्यूपलचे घटक ठेवले पाहिजेत.

उदाहरणार्थ

EMP टेबलवरून शर्माची नोकरी प्रदर्शित करा:

```
DECLARE
  Sjob EMP.Job%type;
BEGIN
  Select Job into Sjob
  From EMP
  Where LastName='Sharma';
```

```

Dbms_output.put_line ('Job of Sharma is ' || Sjob);
END;
/
SQL>/
Job of Sharma is Developer
PL/SQL procedure successfully completed.

```

4.1.5 Constants

- स्थिरांक हे PL/SQL ब्लॉकमध्ये वापरलेले मूल्य आहे जे संपूर्ण प्रोग्राममध्ये अपरिवर्तित राहते.
- स्थिरांक एक मूल्य धारण करतो जे एकदा घोषित केले की, प्रोग्राममध्ये बदलत नाही. एक स्थिर घोषणा त्याचे नाव, डाटा प्रकार आणि मूल्य निर्दिष्ट करते आणि त्यासाठी स्टोरेजचे वाटप करते.
- स्थिरांक हे वापरकर्ता-परिभाषित शाब्दिक मूल्य आहे. तुम्ही स्थिरांक घोषित करू शकता आणि वास्तविक मूल्याऐवजी ते वापरू शकता.
- स्थिरांक घोषित करण्यासाठी सामान्य वाक्यरचना:

```

constant_name CONSTANT datatype := VALUE;
constant_name हे स्थिरांकाचे नाव आहे, म्हणजे व्हेरिएबलच्या नावाप्रमाणेच.
CONSTANT हा शब्द एक कीवर्ड आहे जो त्याचे मूल्य बदलत नाही.

```

VALUE एक मूल्य जे घोषित केले जाते तेव्हा स्थिरांकास नियुक्त करणे आवश्यक आहे.

उदाहरणार्थ:

```

PI CONSTANT NUMBER := 3.141592654;
DECLARE
    -- constant declaration
    pi constant number := 3.141592654;
    -- other declarations
    radius number(5,2);
    dia number(5,2);
    circumference number(7, 2);
    area number (10, 2);
BEGIN
    -- processing
    radius := 9.5;
    dia := radius * 2;
    circumference := 2.0 * pi * radius;
    area := pi * radius * radius;
    -- output
    dbms_output.put_line('Radius: ' || radius);
    dbms_output.put_line('Diameter: ' || dia);
    dbms_output.put_line('Circumference: ' || circumference);
    dbms_output.put_line('Area: ' || area);
END;
/
Output:
Radius: 9.5
Diameter: 19

```

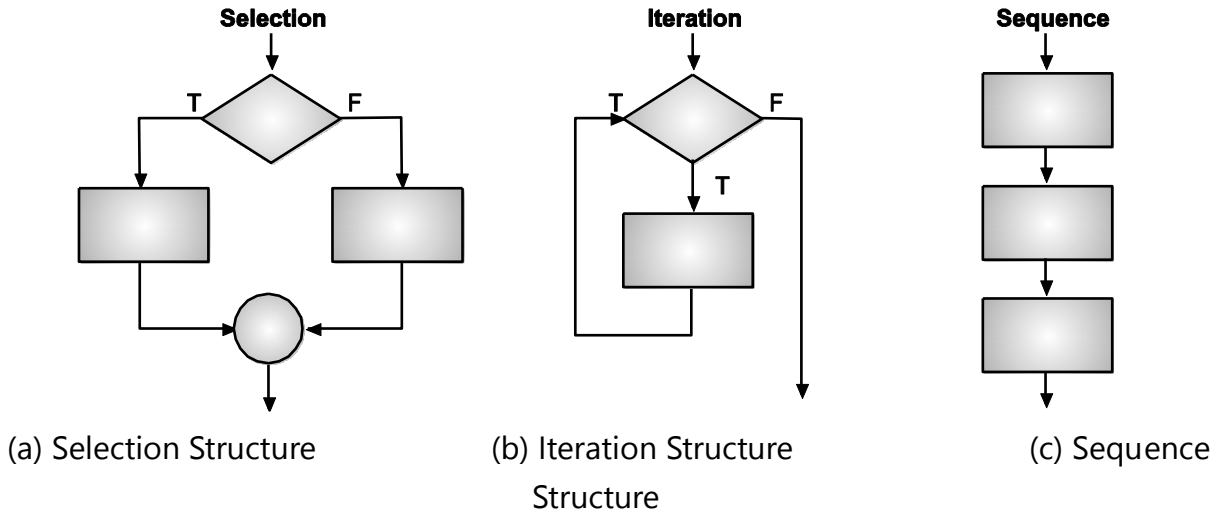
Circumference: 59.69

Area: 283.53

PL/SQL procedure successfully completed.

4.2 नियंत्रण संरचना

- Oracle PL/SQL मध्ये, प्रोग्राममधील अंमलबजावणीचा प्रवाह व्यवस्थापित करण्यासाठी कंट्रोल स्ट्रक्चर्सचा वापर केला जातो. या संरचना विकासकांना विशिष्ट अटी किंवा लूपच्या आधारे विधाने अंमलात आणल्याचा क्रम ठरवू देतात.
- संरचनेच्या प्रमेयानुसार, आकृती 4.3 मध्ये दर्शविलेल्या मूलभूत नियंत्रण संरचनांचा वापर करून कोणताही संगणक प्रोग्राम लिहिला जाऊ शकतो.



आकृती. 4.3

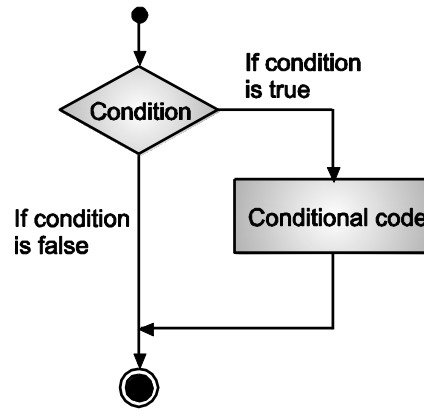
निवड रचना एका अटीची चाचणी घेते, नंतर अट सत्य किंवा असत्य यावर अवलंबून, दुसऱ्याऐवजी विधानांचा एक क्रम कार्यान्वित करते. कंडिशन म्हणजे कोणतेही व्हेरिबल किंवा एक्सप्रेशन जे बूलियन व्हॅल्यू (TRUE किंवा FALSE) मिळवते. पुनरावृत्ती रचना जोपर्यंत एक अट सत्य आहे तोपर्यंत विधानांचा क्रम वारंवार कार्यान्वित करते. अनुक्रम रचना ही विधाने ज्या क्रमाने घडतात त्या क्रमाने कार्यान्वित करते.

4.2.1 Conditional Control

सशर्त नियंत्रण विधानांचे तीन प्रकार आहेत जसे की IF-THEN, IF-THEN-ELSE आणि IF-THEN-ELSIF.

4.2.1.1 IF-THEN विधान:

- IF कंट्रोल स्टेटमेंट, निर्णय घेण्यासाठी आणि कार्यक्रम अंमलबजावणीचा नियंत्रण प्रवाह बदलण्यासाठी वारंवार वापरला जातो.
- IF स्टेटमेंटचा सर्वात सोपा फॉर्म THEN आणि END IF या कीवर्डद्वारे संलग्न केलेल्या विधानांच्या अनुक्रमासह अट संबद्ध करतो.
- जर अट सत्य असेल, तर विधाने कार्यान्वित होतील आणि जर अट FALSE किंवा NULL असेल तर IF विधान काहीही करत नाही.
- जर बूलियन एक्सप्रेशन कंडिशनचे मूल्यमापन खरे असेल तर जर-तर कोडचा ब्लॉक कार्यान्वित केला जाईल अन्यथा कोडचा इतर ब्लॉक कार्यान्वित केला जाईल.
- IF स्टेटमेंटचा फ्लो डायग्राम आकृती 4.4 मध्ये दर्शविला आहे.



आकृती. 4.4

IF_THEN वाक्यरचना :

```

IF <condition> THEN
  <statement_list>
END IF;
  
```

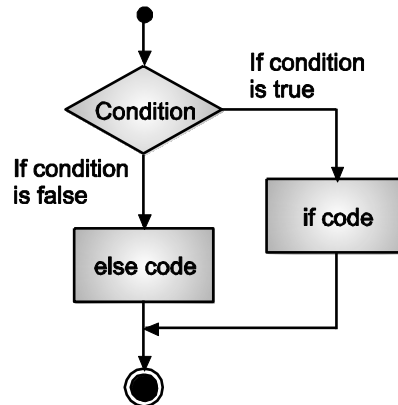
4.2.1.2 IF-THEN-ELSE विधान

- IF-THEN विधानांचा क्रम नंतर ELSE विधानांचा पर्यायी क्रम असू शकतो, जो अटी FALSE असताना कार्यान्वित होतो. IF-THEN-ELSE विधान तुम्हाला विधानांचा क्रम सशर्त कार्यान्वित करू देते.
- IF खंड अट तपासतो; अट सत्य असल्यास काय करावे हे THEN कलम परिभाषित करते; ELSE क्लॉज अट चुकीची किंवा शून्य असल्यास काय करावे हे परिभाषित करते.
- IF THEN ELSE वाक्यरचना खाली दिली आहे

```

IF <condition> THEN
  <statement_list>
ELSE
  <statement_list>
END IF;
  
```

प्रवाह आकृती आकृती 4.5 मध्ये दर्शविली आहे.



आकृती 4.5

तुम्हाला मल्टीवे शाखा हवी असल्यास, तुम्ही IF_THEN_ELSEIF स्टेटमेंट वापरू शकता

4.2.1.3 IF-THEN-ELSEIF विधान

- IF-THEN-ELSIF विधान तुम्हाला अनेक पर्यायांमधून निवडण्याची परवानगी देते. IF-THEN विधान नंतर पर्यायी ELSIF...ELSE विधान केले जाऊ शकते. ELSIF खंड तुम्हाला अतिरिक्त अटी जोडू देतो.
- IF-THEN-ELSIF विधाने वापरताना काही मुद्दे लक्षात ठेवावेत.

1. त्याचा ELSIF ELSEIF नाही.
 2. IF-THEN विधानात शून्य किंवा एक ELSE असू शकते आणि ते कोणत्याही ELSEIF च्या नंतर आले पाहिजे.
 3. IF-THEN विधानामध्ये अनेक ELSIF चे शून्य असू शकतात आणि ते ELSE च्या आधी येणे आवश्यक आहे.
 4. एकदा, ELSIF यशस्वी झाल्यानंतर, उर्वरित ELSIF किंवा ELSE पैकी कोणाचीही चाचणी केली जाणार नाही.
- IF-THEN-ELSEIF वाक्यरचना आहे:


```
IF <condition_1> THEN
  <statement_lists>
ELSIF <condition_2> THEN
  <statement_lists>
... ..
ELSIF <condition_n> THEN
  <statement_lists>
ELSE ...
END IF;
```

प्रोग्राम

प्रोग्राम 1 : दोन संख्यांपैकी सर्वात मोठी संख्या शोधण्यासाठी प्रोग्राम.

```
DECLARE
A NUMBER;
B NUMBER;
BEGIN
A:=&a;
B:=&b;
If (A>B) THEN
dbms_output.put_line ('A IS LARGEST');
ELSE
Dbms_output.put_line ('B IS LARGEST');
END IF;
END;
/
SQL>/
Enter value for A:6
Old 5: A:=&a;
New 5 : A:=6
Enter value for B:7
Old 6: B:=&b
New 5 : B:=7
B IS LARGEST
PL/SQL procedure successfully completed.
```

प्रोग्राम 2: जॉन्सनचा पगार 50000 आहे की नाही हे प्रदर्शित करण्याचा प्रोग्राम.

```

DECLARE
Jsal EMP.Sal%type;
BEGIN
SELECT Sal INTO Jsal
From EMP
Where LastName='Johnson';
IF(Jsal>50000) Then
dbms_output.put_line ('Salary of Johnson is 50000');
ELSE
dbms_output.put_line ('Salary of Johnson is not 50000');
END IF;
END;
/
SQL>/
Salary of Johnson is 50000
PL/SQL procedure successfully completed.

```

प्रोग्राम 3: निव्वळ पगाराच्या गणनेसाठी प्रोग्राम.

```

DECLARE
ename varchar2(15); /*size of ename is 15*/
basic number;
da number;
hra number;
pf number;
netsalary number;
BEGIN
ename:=&ename;
basic:=&basic;
da:=basic * (41/100);
hra:=basic * (15/100);
IF (basic < 3000) THEN
pf:=basic * (5/100);
ELSIF (basic >= 3000 and basic <= 5000) THEN
pf:=basic * (7/100);
ELSIF (basic >= 5000 and basic <= 8000) THEN
pf:=basic * (8/100);
ELSE
pf:=basic * (10/100);
END IF;
netsalary:=basic + da + hra -pf;

```

```

dbms_output.put_line('Employee name : ' || ename);
dbms_output.put_line('Providend Fund : ' || pf);
dbms_output.put_line('Net salary : ' || netsalary);
end;

```

प्रोग्राम 4 :तीन संख्यांमध्ये सर्वात मोठी संख्या शोधण्याचा प्रोग्राम

```

DECLARE
a number;
b number;
c number;
d number;
BEGIN
dbms_output.put_line('Enter a:');
a:=&a;
dbms_output.put_line('Enter b:');
b:=&b;
dbms_output.put_line('Enter c:');
c:=&c;
IF (a>b) and (a>c) THEN
dbms_output.putline('A is Maximum');
ELSIF (b>a) and (b>c) THEN
dbms_output.putline('B is Maximum');
ELSE
dbms_output.putline('C is Maximum');
END IF;
END;

```

4.2.2 पुनरावृत्ती नियंत्रण

- पुनरावृत्ती नियंत्रण विधाने वापरली जातात जेव्हा आपण एक किंवा अधिक विधानांची अंमलबजावणी निर्दिष्ट वेळेसाठी पुनरावृत्ती करू इच्छितो.
- PL/SQL मध्ये तीन प्रकारचे लूप आहेत म्हणजे लूप, फॉर लूप आणि व्हेअर लूप.

1. लूप:

- LOOP स्टेटमेंट्स तुम्हाला स्टेटमेंट्सचा क्रम अनेक वेळा कार्यान्वित करू देतात.
- तुम्ही अनुक्रमातील पहिल्या विधानाच्या आधी LOOP कीवर्ड आणि अनुक्रमातील शेवटच्या विधानानंतर कीवर्ड END LOOP ठेवता.
- वाक्यरचना खाली दिली आहे:

```

LOOP
<loop_body> /* A list of statements. */
END LOOP;

```

- EXIT-WHEN स्टेटमेंट तुम्हाला पुढील प्रक्रिया अशक्य असल्यास लूप पूर्ण करू देते.
- जेव्हा EXIT विधान समोर येते, तेव्हा WHEN खंडातील स्थितीचे मूल्यमापन केले जाते. कंडिशन सत्य असल्यास, लूप पूर्ण होतो आणि नियंत्रण पुढील विधानाकडे जाते.

प्रोग्राम 1 टेबल T1 मध्ये (100, 100) पर्यंत प्रत्येक जोडी (1, 1) समाविष्ट करण्याचा कार्यक्रम.

```
DECLARE
i NUMBER := 1;
BEGIN
LOOP
INSERT INTO T1 VALUES(i,i);
i := i+1;
EXIT WHEN i>100;
END LOOP;
END;
run;
```

प्रोग्राम 2: 1 ते 5 अंक मुद्रित करण्याचा प्रोग्राम 1.

```
DECLARE
i NUMBER:=0;
BEGIN
LOOP
I:=i+1;
Dbms_output.put_line(i);
IF(i>=5) THEN
EXIT;
END IF;
END LOOP;
END;
/
SQL>/
1
2
3
4
5
```

प्रोग्राम 3: EXIT WHEN कंडिशन वापरून 1 ते 5 अंक मुद्रित करण्याचा प्रोग्राम.

```
DECLARE
i NUMBER:=0;
BEGIN
LOOP
i:=i+1;
Dbms_output.put_line(i);
EXIT when i>=5;
END LOOP;
END;
/
SQL>/
1
```

2
3
4
5

2. While Loop:

- WHILE LOOP स्टेटमेंट स्टेटमेंटच्या क्रमाशी अट संबद्ध करते. लूपच्या प्रत्येक पुनरावृत्तीपूर्वी, स्थितीचे मूल्यांकन केले जाते. जर कंडिशन सत्य असेल, तर विधानांचा क्रम कार्यान्वित केला जातो, नंतर लूपच्या शीर्षस्थानी नियंत्रण पुन्हा सुरू होते. कंडिशन असत्य किंवा शून्य असल्यास, लूप बायपास केला जातो आणि कंट्रोल पुढील विधानाकडे जातो.
- वाक्यरचना खाली दिली आहे:

```
WHILE <condition>
LOOP
<loop_body> /*sequence of statements*/
END LOOP;
```

प्रोग्राम 1: 1 ते 5 अंक मुद्रित करण्याचा प्रोग्राम.

```
DECLARE
    i number:=0;
BEGIN
WHILE i<=5 LOOP
    i:=i+1;
    dbms_output.put_line(i);
END LOOP;
END;
/
SQL>/
1
2
3
4
5
```

प्रोग्राम 2: 100 विषम संख्यांची बेरीज शोधण्याचा प्रोग्राम.

```
DECLARE
n NUMBER;
value NUMBER;
sum NUMBER default 0;
BEGIN
value:=&value;
n:=1;
WHILE (n < value)
LOOP
sum:=sum+n;
n:=n+2;
END LOOP;
dbms_output.put_line('Sum of odd numbers between 1 and ' || endvalue || ' is ' ||
```

```
sum);
END;
/
```

3. FOR लूप

- फॉर लूप स्टेटमेंट तुम्हाला पूर्णांकांची श्रेणी निर्दिष्ट करू देते, नंतर श्रेणीतील प्रत्येक पूर्णांकासाठी एकदा विधानांचा क्रम कार्यान्वित करू देते.
- वाक्यरचना खाली दिली आहे

```
FOR i in 1..n LOOP
<loop_body> /* sequence of statements */
END LOOP;
```

प्रोग्राम 1: 1 ते 5 अंक मुद्रित करण्याचा प्रोग्राम.

```
DECLARE
N NUMBER:=5;
FOR I in 1..5 LOOP
dbms_output.put_line(i);
END LOOP;
END;
/
SQL>
1
2
3
4
5
```

प्रोग्राम 2: विषम संख्यांची बेरीज शोधण्यासाठी प्रोग्राम

```
declare
n Number;
sum number default 0;
value Number;
begin
value:=&value;
begin
value:=&value;
n:-1;
for n in 1.. value LOOP
IF mod(n,2)=1 THEN
sum:=sum+n;
END IF;
END LOOP;
dbms_output.put_line('sum='||
DECLARE
n NUMBER;
sum NUMBER default 0;
```

```

value NUMBER;
BEGIN
value:=&value;
n:=1;
for n in 1.. value LOOP
IF mod(n,2)=1 THEN
sum:=sum+n;
END IF;
END LOOP;
dbms_output.put_line('sum = ' || sum);
END;
/

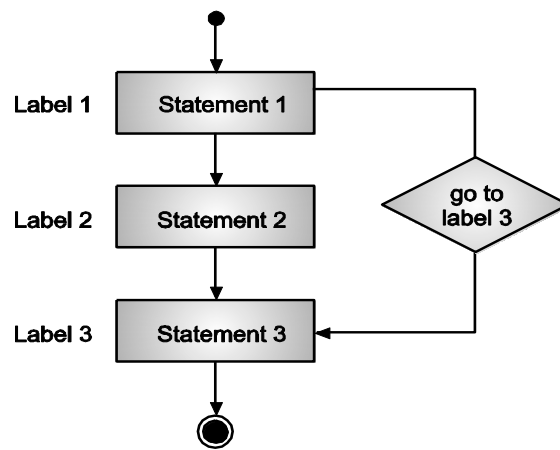
```

4.2.3 Sequential Control

- Oracle PL/SQL GOTO स्टेटमेंट आणि NULL स्टेटमेंट ही ओरॅकलमध्ये उपलब्ध असलेली अनुक्रमिक नियंत्रण रचना आहे.

1. GOTO विधान:

- GOTO स्टेटमेंट तात्काळ प्रोग्राम कंट्रोल ("शाखा" म्हणतात) बिनशर्त नामांकित स्टेटमेंट लेबल किंवा ब्लॉक लेबलवर हस्तांतरित करते. विधान किंवा लेबलचे नाव ब्लॉकमध्ये अद्वितीय असणे आवश्यक आहे.
- GOTO स्टेटमेंट IF स्टेटमेंट, लूप स्टेटमेंट किंवा सब-ब्लॉकमध्ये शाखा करू शकत नाही. GOTO लेबलने एक्झिक्युटेबल स्टेटमेंट किंवा PL/SQL ब्लॉक केले पाहिजे.
- जर GOTO चा वापर कर्सर फॉर लूपमधून वेळेपूर्वी बाहेर पडण्यासाठी केला गेला, तर कर्सर आपोआप बंद होईल.



आकृती 4. 6

सामान्य वाक्यरचना:

```

BEGIN
...
GOTO insert_row;
...
<<insert_row>>
INSERT INTO emp VALUES ...
END;
उदाहरण

```



```

DECLARE
  v_last_name VARCHAR2(25);
  v_emp_id    NUMBER(6) := 120;
BEGIN
  <<get_name>>
  SELECT last_name INTO v_last_name
  FROM employees
  WHERE employee_id = v_emp_id;
  BEGIN
    DBMS_OUTPUT.PUT_LINE (v_last_name);
    v_emp_id := v_emp_id + 5;

    IF v_emp_id < 120 THEN
      GOTO get_name;
    END IF;
  END;
END;
/

```

2. NULL विधान:

NULL स्टेटमेंट काहीही करत नाही आणि पुढील स्टेटमेंटवर नियंत्रण देते. काही भाषा अशा सूचनांना नो-ऑप (नो ऑपरेशन) म्हणून संबोधतात.

उदाहरण

```

DECLARE
  v_job_id VARCHAR2(10);
  v_emp_id NUMBER(6) := 110;
BEGIN
  SELECT job_id INTO v_job_id FROM employees WHERE employee_id=
  v_emp_id;
  IF v_job_id = 'SA_REP' THEN
    UPDATE employees SET commission_pct = commission_pct * 1.2;
  ELSE
    NULL; -- do nothing if not a sales representative
  END IF;
END;
/

```

4.3 Exception Handling

- अपवाद एक त्रुटी परिस्थिती आहे, जी प्रोग्रामच्या अंमलबजावणी दरम्यान उद्भवते. जेव्हा एखादी त्रुटी येते तेव्हा अपवाद वाढविला जातो, सामान्य अंमलबजावणी थांबविली जाते आणि नियंत्रण अपवाद-हँडलिंग भागाकडे हस्तांतरित केले जाते.
- अपवाद हँडलर हे अपवाद हाताळण्यासाठी लिहिलेले नित्यक्रम आहेत.
- PL/SQL अपवाद हाताळण्यासाठी एक वैशिष्ट्य प्रदान करते जे अपवाद हाताळणी म्हणून ओळखल्या जाणाऱ्या PL/SQL ब्लॉकमध्ये आढळतात.

- अपवाद हाताळणी वापरून आपन कोडची चाचणी करू शकतो आणि तो अनपेक्षितपणे बाहेर पडण्यापासून टाळू शकतो. जेव्हा एखादा अपवाद येतो तेव्हा त्याचे कारण स्पष्ट करणारा संदेश प्राप्त होतो.
- अपवाद जोडण्यासाठी वाक्यरचना खाली दिली आहे:

```

DECLARE
Declaration section
BEGIN
Exception section
EXCEPTION
WHEN ex_name1 THEN
-Error handling statements
WHEN ex_name2 THEN
-Error handling statements
WHEN Others THEN
-Error handling statements
END;
```

जेव्हा अपवाद केला जातो, तेव्हा Oracle अपवाद विभागात योग्य अपवाद हँडलर शोधते.

4.3.1 पूर्वनिर्धारित अपवाद किंवा नामित प्रणाली अपवाद

- जेव्हा एखादा प्रोग्राम RDBMS नियमाचे उल्लंघन करतो तेव्हा Oracle द्वारे सिस्टम अपवाद आपोआप उठवले जातात.
- काही सिस्टीम अपवाद आहेत जे वारंवार उठवले जातात, म्हणून ते पूर्व-परिभाषित आहेत आणि ओरॅकलमध्ये नाव दिले आहे जे नामित सिस्टम अपवाद म्हणून ओळखले जाते.

उदाहरणार्थ NO_DATA_FOUND आणि ZERO_DIVIDE यांना नामांकित सिस्टम अपवाद म्हणतात

- नामांकित प्रणाली अपवाद आहेत:
 - स्पष्टपणे घोषित नाही,
 - पूर्वनिर्धारित ओरॅकल त्रुटी उद्भवते तेव्हा अस्पष्टपणे वाढविले जाते.

क्र.	अपवाद नाव	कारण
1	CURSOR_ALREADY_OPEN	जेव्हा तुम्ही कर्सर उघडता जो आधीच उघडलेला असतो
2	INVALID_CURSOR	तुम्ही कर्सर बंद करण्यासारखे कर्सरवर अवैध ऑपरेशन करता तेव्हा
3	NO_DATA_FOUND	जेव्हा SELECT...INTO क्लॉज टेबलमधून कोणतीही पंक्ती देत नाही.
4	TOO_MANY_ROWS	जेव्हा तुम्ही रेकॉर्डमध्ये एकापेक्षा जास्त पंक्ती निवडता किंवा आणता.

- ज्या सिस्टीम अपवादासाठी ओरॅकल नाव देत नाही त्यांना अनमेड सिस्टम अपवाद म्हणून ओळखले जाते. हे अपवाद वारंवार होत नाहीत. या अपवादांमध्ये एक कोड आणि संबंधित संदेश असतो.
- अनामित प्रणाली अपवाद हाताळण्याचे दोन मार्ग आहेत:
 - WHEN OTHERS अपवाद हँडलर वापरून, किंवा
 - अपवाद कोड नावाशी जोडून आणि नामित अपवाद म्हणून वापरून.

उदाहरण

```

DECLARE
N number;
BEGIN
N:=10/0;
EXCEPTION
WHEN ZERO_DIVIDE THEN
Dbms_output.put_line('Zero Divide error');
END;
/

Zero Divide error
PL/SQL procedure successfully completed.
DECLARE
Tsal number(7,2);
BEGIN
Select salary INTO Tsal from emp where empno=1111;
If Tsal>5000 then
Update emp SET salary=Tsal*2.00 where empno>=1111;
Else
Update emp SET salary=10000 where empno>=1111;
End if;
EXCEPTION
WHEN NO_DATA_FOUND THEN
Dbms_output.put_line('Empno does not exist');
WHEN TOO_MANY_ROWS THEN
Dbms_output.put_line('No. of rows selected');
END;
/

```

4.3.2 वापरकर्ता परिभाषित अपवाद

वापरकर्ता-परिभाषित अपवाद प्रोग्रामरद्वारे परिभाषित करणे आवश्यक आहे. वापरकर्ता-परिभाषित अपवाद घोषणा विभागात त्यांच्या प्रकारासह अपवाद म्हणून घोषित केले जातात.

वापरकर्ता परिभाषित अपवाद परिभाषित करण्यासाठी दोन पद्धती आहेत

1. RAISE, आणि
2. RAISE_APPLICATION_ERROR

ते RAISE स्टेटमेंट वापरून स्पष्टपणे उभे केले जाणे आवश्यक आहे, पूर्व-परिभाषित अपवादांच्या विपरीत जे अस्पष्टपणे उठवले जातात.

अपवाद घोषित करणे:

```

DECLARE
exception_name EXCEPTION;
BEGIN
---statements---
Raising Exception:
BEGIN

```

```
RAISE exception_name;
```

```
---statements----
```

Handling Exception:

```
BEGIN
```

```
-----
```

```
-----
```

```
EXCEPTION
```

```
WHEN exception_name THEN
```

```
Statements;
```

```
END;
```

- तुमचे स्वतःचे त्रुटी संदेश प्रदर्शित करण्यासाठी अंगभूत RAISE_APPLICATION_ERROR वापरू शकतो. ते ओरॅकल त्रुटीप्रमाणेच त्रुटी संदेश प्रदर्शित करतात.
- error_number साठी -20000 ते -20999 दरम्यान ऋण संख्या वापरा आणि त्रुटी संदेश 512 वर्णपेक्षा जास्त नसावा.
- raise_application_error कॉल करण्यासाठी सिटॅक्स आहे,
- RAISE_APPLICATION_ERROR (error_number, error_message, [TRUE | FALSE]);

उदाहरणे

1. पगार नकारात्मक आहे की नाही हे तपासण्यासाठी. नकारात्मक असल्यास अपवाद वाढवा.

```
DECLARE
Tsal number(7,2);
NEGATIVE_SALARY EXCEPTION;
BEGIN
Select salary INTO Tsal from emp where empno=1111;
If tsal<0 then
Raise NEGATIVE_SALARY;
ELSE
Update emp set salary=10000 where empno=1111;
End if;
EXCEPTION
WHEN NO_DATA_FOUND THEN
Dbms_output.put_line('Record not found');
WHEN NEGATIVE_SALARY THEN
Dbms_output.put_line('Salary is Negative');
END;
```

2. emp चा पगार अपडेट करण्यासाठी आणि empno 1111 सह emp आढळला नाही तर अपवाद वाढवा.

```
DECLARE
Tsal number(7,2);
BEGIN
Select salary INTO Tsal from emp where empno=1111;
Update emp set salary=10000 where empno=1111;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20100,'Record not found');
```

END;

4.4 कर्सर (cursor)

- PL/SQL कर्सर हा एक पॉइंटर आहे जो डाटाबेस टेबलच्या विरूद्ध SQL क्वेरीच्या निकालांच्या सेटकडे निर्देश करतो. कर्सर हे PL/SQL मधील एक कार्य क्षेत्र आहे जे क्वेरीचा परिणाम संचयित करण्यासाठी वापरल्या जाणाऱ्या sql सर्व्हरद्वारे वापरले जाते.
- प्रत्येक स्तंभ मूल्य पॉइंटर वापरून पॉइंट केले आहे. आपण स्वतंत्रपणे कर्सर मूल्ये हाताळू शकता. ते कार्य करत आहे याबद्दल थोडेसे, समजा तुम्ही सर्व्हरमध्ये संग्रहित केलेली क्वेरी विचारली. सुरुवातीला क्वेरी परिणामांचा समावेश असलेला कर्सर सर्व्हरमध्ये तयार केला जातो...
- आता कर्सर क्लायंटकडे हस्तांतरित केला जातो जेथे पुन्हा कर्सर तयार केला जातो आणि त्यामुळे परिणाम प्रदर्शित होतो.
- दोन प्रकारचे कर्सर आहेत:
 1. इम्प्लिसिट
 2. एक्स्प्लिसिट.

PL/SQL सर्व SQL डाटा मॅनिपुलेशन स्टेटमेंटसाठी कर्सर स्पष्टपणे घोषित करते, ज्यामध्ये फक्त एक पंक्ती परत करणाऱ्या क्वेरीचा समावेश आहे. एकापेक्षा जास्त पंक्ती परत करणाऱ्या क्वेरीसाठी, तुम्ही स्वतंत्रपणे पंक्तींवर प्रक्रिया करण्यासाठी कर्सर स्पष्टपणे घोषित करू शकता.

4.4.1 इम्प्लिसिट आणि एक्स्प्लिसिट कर्सर (implicit and explicit cursor)

दोन प्रकारचे कर्सर आहेत, इम्प्लिसिट कर्सर आणि एक्स्प्लिसिट कर्सर.

1. इम्प्लिसिट कर्सर (Implicit Cursor)

- जेव्हा डीएमएल स्टेटमेंट, INSERT, UPDATE आणि DELETE स्टेटमेंट्स कार्यान्वित केल्या जातात तेव्हा ते डीफॉल्टनुसार तयार केले जातात. जेव्हा फक्त एक पंक्ती मिळवणारे SELECT स्टेटमेंट कार्यान्वित केले जाते तेव्हा ते देखील तयार केले जातात.
- जेव्हा तुम्ही DELETE, INSERT, UPDATE आणि SELECT स्टेटमेंट्स सारखी DML स्टेटमेंट्स कार्यान्वित करता, तेव्हा या स्टेटमेंट्सवर प्रक्रिया करण्यासाठी अंतर्निहित स्टेटमेंट्स तयार केली जातात. Oracle DML ऑपरेशन्सची स्थिती तपासण्यासाठी अंतर्निहित कर्सर विशेषता म्हणून ओळखल्या जाणाऱ्या काही विशेषता प्रदान करते.

SQL कर्सर विशेषता :

1. %ROWCOUNT: SQL विधानाद्वारे प्रक्रिया केलेल्या पंक्तींची संख्या.
2. %FOUND: किमान एका पंक्तीवर प्रक्रिया केली असल्यास खरे.
3. %NOTFOUND: कोणत्याही पंक्तीवर प्रक्रिया केली नसल्यास खरे.
4. %ISOPEN: कर्सर उघडल्यास खरे किंवा कर्सर उघडले नसल्यास किंवा बंद केले असल्यास असत्य. केवळ इम्प्लिसिट कर्सर सह वापरले जातो

1. EMP टेबलमधून हटवलेल्या पंक्तींची संख्या मुद्रित करण्यासाठी प्रोग्राम.

```

DECLARE
ROW_DEL NUMBER;
BEGIN
DELETE FROM EMP;
ROW_DEL:=SQL%ROWCOUNT;
DBMS_OUTPUT.PUT_LINE('No of rows deleted are'|| ROW_DEL);
End;
```

/

SQL>/

No of rows deleted are 5

PL/SQL procedure successfully completed.

2. टेबल EMP मध्ये एक पंक्ती इनपुट करण्यासाठी आणि तपशील मुद्रित करण्यासाठी प्रोग्राम.

DECLARE

NEW_NO EMP.EmpNo%TYPE:=&NEW_NO;

NEW_NAME EMP.LastName%TYPE;

NEW_JOB EMP.Job%type;

NEW_DEPTNO EMP.DeptNo%TYPE;

BEGIN

SELECT EmpNo,LastName,Job,DeptNo FROM EMP WHERE EmpNo=NEW_NO;

IF SQL%FOUND THEN

DBMS_OUTPUT.PUT_LINE(NEW_NO||' '||NEW_NAME||' '||NEW_JOB||' '||NEW_DEPTNO);

EXCEPTION

WHEN

NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('EmpNo does not exist');

END;

/

SQL>/

Enter value for NEW_NO:7499

7499 Mante Tester 30

PL/SQL procedure successfully completed

एक्स्प्लिसिट कर्सर (Explicit Cursor)

- जर PL/SQL मधील निवडक विधाने अनेक पंक्ती देत असतील तर तुम्हाला एक स्पष्ट कर्सर तयार करावा लागेल.
- बहु-पंक्ती क्वेरीद्वारे परत केलेल्या पंक्तींच्या संचाला परिणाम संच म्हणतात. त्याचा आकार आपल्या शोध निकषांची पूर्तता करणाऱ्या पंक्तींची संख्या आहे.

एक्स्प्लिसिट कर्सरचे वाक्यरचना:

Cursor cursorname IS sql select statement;

एक्स्प्लिसिट कर्सर खालील तीन नियंत्रण विधाने वापरून नियंत्रित केला जाऊ शकतो.

1. OPEN : ओपन स्टेटमेंट सक्रिय संच ओळखते...उदा. निवड विधानाद्वारे क्वेरी परत केली.**ओपन कर्सरचे वाक्यरचना:**

open cursorname;

2. FETCH: फेच स्टेटमेंट व्हेरिएबल्समध्ये पंक्ती आणते. लूप आणि फेच स्टेटमेंटसाठी कर्सर वापरून कर्सर वापरता येतात.**फेच वाक्यरचना:**

fetch cursorname into variable1, variable2...;

3. CLOSE: क्लोज स्टेटमेंट कर्सर बंद करते.

क्लोजचे वाक्यरचना:

```
close cursorname;
```

एक्स्प्लिसिट कर्सर घोषित करण्यासाठी सामान्य वाक्यरचना आहे:

```
DECLARE
variables;
records;
create a cursor;
BEGIN
OPEN cursor;
FETCH cursor;
process the records;
CLOSE cursor;
END;
```

1. कर्सर वापरून पंक्तींमध्ये प्रवेश करण्यासाठी प्रोग्राम.

```
DECLARE
CURSOR c1 is SELECT * FROM EMP;
str_empno EMP.EmpNo%type;
str_ename EMP.LastName%type;
str_job EMP.Job%type;
str_mgr EMP.Mgr%type;
str_hirdate EMP.Hirdate%type;
str_sal EMP.Sal%type;
str_deptno EMP.DeptNo%type;
rno number;
BEGIN
rno := &rno;
FOR e_rec IN c1
LOOP
IF c1%rowcount = rno THEN
DBMS_OUTPUT.PUT_LINE (str_empno || ' ' || str_ename || ' ' || str_job || ' ' ||
str_mgr || ' ' || str_hirdate || ' ' || str_sal || ' ' || str_deptno);
END IF;
END LOOP;
END;
```

2. सारणीचे मूल्य प्रदर्शित करण्यासाठी प्रोग्राम

```
DECLARE
CURSOR c1 IS SELECT * FROM EMP;
e_rec EMP%rowtype;
BEGIN
OPEN c1;
LOOP
FETCH c1 INTO e_rec;
DBMS_OUTPUT.PUT_LINE('Number: ' || ' ' || e_rec.EmpNo);
DBMS_OUTPUT.PUT_LINE('Name : ' || ' ' || e_rec.LastName);
```

```

DBMS_OUTPUT.PUT_LINE('Salary: ' || ' ' || e_rec.Sal);
EXIT WHEN c1%NOTFOUND;
END LOOP;
CLOSE c1;
END;

```

3. सर्वाधिक 10 पगार देणाऱ्या कर्मचाऱ्यांचे तपशील प्रदर्शित करण्यासाठी प्रोग्राम.

```

DECLARE
CURSOR c1 IS SELECT * FROM EMP ORDER BY Sal DESC;
e_rec EMP%rowtype;
BEGIN
FOR e_rec IN c1
LOOP
DBMS_OUTPUT.PUT_LINE('Number: ' || ' ' || e_rec.EmpNo);
DBMS_OUTPUT.PUT_LINE('Name : ' || ' ' || e_rec.LastName);
DBMS_OUTPUT.PUT_LINE('Salary: ' || ' ' || e_rec.Sal);
EXIT WHEN c1%ROWCOUNT >= 10;
END LOOP;
END;

```

4.4.2 कर्सर घोषित करणे, उघडणे आणि बंद करणे

- तुम्ही कर्सर नियंत्रित करण्यासाठी DECLAR, OPEN आणि क्लोज स्टेटमेंट वापरता.
- DECLARE विधान कर्सर व्हेरिएबल घोषित करते.

```

DECLARE
CURSOR cursor_name IS
SELECT statement;

```

- ओपन स्टेटमेंट कर्सरशी संबंधित क्वेरी कार्यान्वित करते, निकाल सेट ओळखते आणि कर्सरला पहिल्या पंक्तीच्या आधी ठेवते.

```
OPEN cursor_name;
```

- जेव्हा शेवटच्या पंक्तीवर प्रक्रिया केली जाते, तेव्हा क्लोज स्टेटमेंट कर्सर अक्षम करते.

```
close cursor_name;
```

- कर्सर उघडल्यावर, पहिली पंक्ती वर्तमान पंक्ती बनते.

4.4.3 कर्सरवरून रेकॉर्ड मिळवा

- कर्सर आणण्यासाठी एका वेळी एका पंक्तीमध्ये प्रवेश करणे समाविष्ट आहे. जेव्हा डाटा प्राप्त केला जातो तेव्हा तो रेकॉर्ड किंवा व्हेरिएबल्समध्ये कॉपी केला जातो आणि लॉजिकल पॉइंटर पुढील पंक्तीकडे जातो आणि ती वर्तमान पंक्ती बनते.
- प्रत्येक फेच स्टेटमेंटवर, पॉइंटर पुढील पंक्तीकडे जातो. जेव्हा कर्सरमध्ये एकापेक्षा जास्त पंक्ती असतात तेव्हा आम्ही सर्व रेकॉर्ड्स आणण्यासाठी स्पष्ट कर्सर विशेषतांसह लूप वापरू शकतो.
- आम्ही कर्सरमधील पंक्ती PL/SQL रेकॉर्ड किंवा PL/SQL ब्लॉकमध्ये तयार केलेल्या व्हेरिएबल्सची सूची मिळवू शकतो. तुम्ही PL/SQL रेकॉर्डवर कर्सर आणत असल्यास, रेकॉर्डची रचना कर्सरसारखीच असावी.
- जर तुम्ही व्हेरिएबल्सच्या सूचीमध्ये कर्सर आणत असाल, तर व्हेरिएबल्स कर्सरमध्ये जसे स्तंभ उपस्थित असतात त्याच क्रमाने फेच स्टेटमेंटमध्ये सूचीबद्ध केले जावेत.

वाक्यरचना खाली दिली आहे:

किंवा
 FETCH cursor_name INTO record_name;
 FETCH cursor_name INTO variable_list;

उदाहरण

```
DECLARE
emp_rec emp_tbl%rowtype;
CURSOR emp_cur IS
SELECT *
FROM emp_tb1
WHERE salary > 10;
BEGIN
OPEN emp_cur;
FETCH emp_cur INTO emp_rec;
dbms_output.put_line (emp_rec.first_name || ' ' || emp_rec.last_name);
CLOSE emp_cur;
END;
```

4.4.4 कर्सर FOR लूप

- बऱ्याच परिस्थितींमध्ये ज्यांना एक्स्प्लिसिट कर्सर आवश्यक आहे, तुम्ही ओपन, फेच आणि क्लोज स्टेटमेंट्सऐवजी कर्सर फॉर लूप वापरून कोडिंग सुलभ करू शकता.
- कर्सर फॉर लूप त्याच्या लूप इंडेक्सला रेकॉर्ड म्हणून घोषित करतो जे डाटाबेसमधून आणलेल्या पंक्तीचे प्रतिनिधित्व करते. पुढे, तो कर्सर उघडतो, रेकॉर्डमधील फील्डमध्ये सेट केलेल्या परिणामांमधून मूल्यांच्या पंक्ती वारंवार आणतो, त्यानंतर सर्व पंक्तींवर प्रक्रिया केल्यावर कर्सर बंद करतो.

उदाहरणार्थ

```
DECLARE
cursor c1 is select * from EMP;
begin
for outvariable in c1
loop
exit when c1%notfound;
if outvariable.Sal < 20000 then
dbms_output.put_line(outvariable.Sal || ' ' || outvariable.LastName);
end if;
end loop;
end;
Cursor for Loop with parameters:
Declare
Cursor c1(dno number) is select * from EMP where DeptNo = dno;
begin
for emprec in c1(10) loop;
dbms_output.put_line(emprec.LastName);
```

```
end loop;
end;
```

4.4.5 पॅरामीटराइज्ड कर्सर

- PL/SQL पॅरामीटराइज्ड कर्सर पॅरामीटर्स कर्सरमध्ये पास करतो आणि त्यांचा केरीमध्ये वापर करतो.
- डीफॉल्ट मूल्ये कर्सर पॅरामीटर्सना नियुक्त केली जातात. आणि पॅरामीटर्सची व्याप्ती स्थानिक पातळीवर आहे.
- PL/SQL पॅरामीटराइज्ड कर्सर फक्त डाटाटाइप परिभाषित करतो.

उदाहरणार्थ

ज्याचा emp_id 10 आहे त्या Emp सारणीवरून कर्मचारी माहिती कर्सर प्रदर्शित करा.

```
DECLARE
    Cursor c(n number) is select* from Emp where emp_id = n;
    Emp1 Emp% rowtype;
BEGIN
    Open c(10);
    For Emp1 IN c(10) LOOP
        dbms_output.put_line ('emp_id'|| Emp1.emp_id);
        dbms_output.put_line (emp_name:'| Emp1.emp_name);
        dbms_output.put_line ('Salary'|| Emp1.salary);
    END Loop;
    Close c;
END;
```

4.5 Procedures

- एक संग्रहित **Procedures** किंवा साध्या भाषेत proc नावाचा PL/SQL ब्लॉक आहे जो एक किंवा अधिक विशिष्ट कार्य करतो. हे इतर प्रोग्रामिंग भाषांमधील प्रक्रियेसारखेच आहे.

4.5.1 फायदे

संचयित **Procedures** विकास, अखंडता, सुरक्षा, कार्यप्रदर्शन आणि मेमरी वाटप या क्षेत्रांमध्ये फायदे देतात.

1. **जलद:** Procedures डाटाबेसमध्ये संग्रहित केल्या जातात आणि त्यामुळे डाटाबेस सर्व्हरवर कार्यान्वित केल्या जातात जे माझ्यासाठी क्लायंटपेक्षा अधिक शक्तिशाली आहेत.
2. **संसाधने जतन करणे:** कोड पूर्व-संकलित स्वरूपात संग्रहित केला जातो ज्याचा अर्थ असा आहे की तो सिंटॅक्टिकली वैध आहे आणि रन-टाइममध्ये संकलित करण्याची आवश्यकता नाही, ज्यामुळे संसाधनांची बचत होते.
3. **सुधारित स्केलेबिलिटी:** संचयित Procedures प्रत्येक वापरकर्ता तंतोतंत समान स्वरूपाचा केरी वापरेल ज्याचा अर्थ पार्सिंग ओव्हरहेड कमी करून आणि अनुप्रयोगांची स्केलेबिलिटी सुधारून केरी पुन्हा वापरल्या जातात.
4. **कमी नेटवर्क रहदारी:** Procedures डाटाबेसमध्ये संग्रहित केल्यामुळे, क्लायंटकडून डाटाबेस सर्व्हरवर कोड हस्तांतरित करण्याची किंवा सर्व्हरवरून क्लायंटकडे मध्यवर्ती निकाल हस्तांतरित करण्याची आवश्यकता नाही.

4.5.2 एक Procedures तयार करणे

- **Procedures** शीर्षलेख आणि मुख्य भाग असतो. कीवर्ड IS च्या आधीच्या भागाला प्रक्रियेचे शीर्षलेख किंवा प्रक्रियेची स्वाक्षरी म्हणतात.

- हेडरमध्ये प्रक्रियेचे नाव आणि प्रक्रियेला दिलेले पॅरामीटर्स किंवा व्हेरिएबल्स असतात. IS या कीवर्ड नंतरची प्रत्येक गोष्ट process's body म्हणून ओळखली जाते.
- मुख्य भागामध्ये सामान्य PL/SQL ब्लॉक प्रमाणेच घोषणा विभाग, अंमलबजावणी विभाग आणि अपवाद विभाग असतो. क्रिएट किंवा रिप्लेस प्रक्रिया विधानासाठी सरलीकृत वाक्यरचना खालीलप्रमाणे आहे:

```
CREATE [OR REPLACE] PROCEDURE procedure_name
[(parameter_name [IN | OUT | IN OUT] type [, ...])]
{IS | AS}
BEGIN
< procedure_body >
END procedure_name;
```

- procedure-name प्रक्रियेचे नाव निर्दिष्ट करते.
- [OR REPLACE] पर्याय विद्यमान कार्यपद्धतीमध्ये बदल करण्यास अनुमती देतो.
- parameter list सूचीमध्ये पॅरामीटर्सचे नाव, मोड आणि प्रकार समाविष्ट आहेत. IN हे मूल्य बाहेरून पास केले जाईल असे दर्शविते आणि OUT दर्शविते की या पॅरामीटरचा वापर प्रक्रियेच्या बाहेरील मूल्य परत करण्यासाठी केला जाईल.
- Procedure-body -बॉडीमध्ये एक्झिक्युटेबल भाग असतो.
- एक स्वतंत्र **Procedures** तयार करण्यासाठी IS कीवर्डऐवजी AS कीवर्ड वापरला जातो.

प्रक्रियांना पॅरामीटर्स पास करणे

PL/SQL मध्ये, आपण तीन प्रकारे प्रक्रियांमध्ये पॅरामीटर्स पास करू शकतो.

1. IN type parameter: या प्रकारच्या पॅरामीटर्सचा वापर संग्रहित प्रक्रियेसाठी मूल्ये पाठवण्यासाठी केला जातो.
2. OUT type parameter: या प्रकारच्या पॅरामीटर्सचा वापर संग्रहित प्रक्रियांमधून मूल्ये मिळविण्यासाठी केला जातो. हे फंक्शन्समधील रिटर्न प्रकारासारखे आहे.
3. IN OUT parameter: या प्रकारच्या पॅरामीटर्सचा उपयोग मूल्ये पाठवण्यासाठी आणि संग्रहित प्रक्रियांमधून मूल्ये मिळविण्यासाठी केला जातो. डीफॉल्टनुसार ते एक IN प्रकार पॅरामीटर आहे.

1. IN Parameter:

- आपण या पॅरामीटर्स किंवा व्हेरिएबल्सद्वारे संग्रहित प्रक्रियेत मूल्ये पास करू शकतो. या प्रकारचे पॅरामीटर हे केवळ वाचनीय पॅरामीटर आहे.
- आपण व्हेरिएबलला IN प्रकार पॅरामीटरचे मूल्य नियुक्त करू शकतो किंवा ते केरीमध्ये वापरू शकतो, परंतु आम्ही प्रक्रियेमध्ये त्याचे मूल्य बदलू शकत नाही.

वाक्यरचना:

```
CREATE [OR REPLACE] PROCEDURE procedure_name (
param_name1 IN datatype, param_name12 IN datatype ... )
param_name1, param_name2... ही अद्वितीय पॅरामीटर नावे आहेत.
```

डाटाटाइप व्हेरिएबलचा डाटाटाइप परिभाषित करतो.

IN पर्यायी आहे, डीफॉल्टनुसार ते IN प्रकार पॅरामीटर आहे.

उदाहरण:

```
CREATE OR REPLACE PROCEDURE proc1(p-no IN number)
IS
sal number(8,2);
BEGIN
```

```

Select salary into sal from emp where empno=p-no;
If sal> 2000 then
Update emp set salary=sal*2.00 where empno=p-no;
Else
Update emp set salary=10000 where empno=p-no;
End if;
END proc1;
SQL>execute proc1(1111);

```

किंवा

प्रक्रिया कार्यान्वित करण्यासाठी खालील कोड लिहा. प्रणाली वापरकर्त्यास empno प्रविष्ट करण्यास सांगते. प्रक्रिया कार्यान्वित आणि अद्यतनित केली आहे की नाही हे पाहण्यासाठी सिलेक्ट स्टेटमेंट वापरून मूळ emp टेबल तपासा.

```

DECLARE
Temp_empno number;
BEGIN
Proc1(&Temp_empno);
END;
/
SQL>/

```

1. OUT Parameter:

- आउट पॅरामीटर्सचा वापर प्रक्रिया किंवा फंक्शनमधून आउटपुट पाठवण्यासाठी केला जातो.
- हे फक्त लिहिण्याजोगे पॅरामीटर आहे म्हणजे, संग्रहित प्रक्रिया राबवताना आम्ही मूल्ये OUT पॅरामीटर्समध्ये पास करू शकत नाही, परंतु आम्ही संचयित प्रक्रियेच्या आत OUT पॅरामीटरला मूल्ये नियुक्त करू शकतो आणि कॉलिंग प्रोग्राम हे आउटपुट मूल्य प्राप्त करू शकतो.
- आउट पॅरामीटर तयार करण्यासाठी सिंटॅक्स:

```
CREATE [OR REPLACE] PROCEDURE proc2 (param_name OUT datatype)
```

उदाहरण: प्रक्रियेसाठी युक्तिवाद म्हणून empno पास करा आणि त्या कर्मचार्याचे नाव व्यवस्थापक प्रदर्शित करा.

```

CREATE OR REPLACE PROCEDURE details(eno IN number,ename OUT varchar2)
AS
Tempname emp.name%type;
BEGIN
Select name into tempname from emp where empno=(select mgr from emp where
empno=eno);
Ename:=tempname;
END details;

```

कॉलिंग प्रोग्राम

```

DECLARE
tno number;
tname varchar2(20);
BEGIN
tno:=&tno;

```

```

details(tno,tname);
if(tname='no')then
Dbms_output.put_line('emp no. does not exist');
Else
Dbms_output.put_line('Name of Manager is '||tname);
End if;
END;
SQL>/

```

1. IN OUT Parameter:

- IN OUT पॅरामीटर आम्हाला प्रक्रियेमध्ये मूल्ये पास करण्यास आणि प्रक्रियेतून आउटपुट मूल्ये प्राप्त करण्यास अनुमती देते. कॉलिंग प्रोग्राममध्ये IN पॅरामीटरचे मूल्य बदलता येत असल्यास हे पॅरामीटर वापरले जाते.
- इन आउट पॅरामीटर वापरून आपण पॅरामीटरमध्ये मूल्ये पास करू शकतो आणि त्याच पॅरामीटरचा वापर करून कॉलिंग प्रोग्रामला मूल्य परत करू शकतो. परंतु हे केवळ तेव्हाच शक्य आहे जेव्हा प्रक्रियेला दिलेले मूल्य आणि आउटपुट मूल्य समान डाटाटाइप असेल.
- प्रक्रियेत पॅरामीटरचे मूल्य बदलल्यास हे पॅरामीटर वापरले जाते.
- वाक्यरचना:
- CREATE [OR REPLACE] PROCEDURE proc3 (param_name IN OUT datatype)
- उदाहरण: प्रक्रियेसाठी विभागाचे नाव पास करा. प्रक्रिया पास होईल क्र. त्या विभागात काम करणाऱ्या कर्मचाऱ्यांची संख्या समान व्हेरिबलमध्ये.

```

CREATE OR REPLACE PROCEDURE Dept(dname IN OUT varchar2)
AS
Deptname varchar2;
BEGIN
Select count(*) into Deptname from staff where dept=dname;
Dname:=Deptname;
END Dept;

```

कॉलिंग प्रोग्राम

```

DECLARE
Dno number;
BEGIN
Dno:=&dno;
Dept(dno);
If(dno=0) then
Dbms_output.put_line('Department does not exist');
Else
Dbms_output.put_line('No. of staff working are '||dno);
End if;
END;
SQL>/

```

4.5.3 कार्यपद्धती अंमलात आणणे आणि हटवणे

प्रक्रिया राबवणे:

प्रक्रिया अंमलात आणण्याचे दोन मार्ग आहेत.

i) SQL प्रॉम्प्ट वरून:

SQL>execute procedurename (argument value);

उदाहरण:

SQL>/execute proc1(20);

ii) कोड लिहा:

DECLARE

declaration

BEGIN

Statements

END;

Example:

DECLARE

Dno number;

BEGIN

Dno:=&dno;

Dept(dno);

If(dno=0) then

Dbms_output.put_line('Department does not exist');

Else

Dbms_output.put_line('No. of staff working are '||dno);

End if;

END;

2. Deleting a Procedure:

प्रक्रिया हटवण्यासाठी DROP PROCEDURE कमांड वापरली जाते.

वाक्यरचना :

DROP PROCEDURE procedurename;

उदाहरण:

DROP PROCEDURE proc1;

4.6 फंक्शन (Function)

- फंक्शन हे SQL आणि PL/SQL स्टेटमेंट्सचा तार्किकदृष्ट्या गटबद्ध संच आहे जे विशिष्ट कार्य करतात. फंक्शन हा PL/SQL ब्लॉक आहे जो ओरॅकल इंजिनच्या सिस्टीम टेबलपैकी एकामध्ये संकलित आणि संग्रहित केला गेला आहे.
- फंक्शन डायनॅमिक बनवण्यासाठी त्यांपैकी एकाला एक्झिक्यूशन करण्यापूर्वी पॅरामीटर्स पास केले जाऊ शकतात. एखादे फंक्शन नंतर त्याच्या कार्यान्वित होण्यापूर्वी पास केलेल्या पॅरामीटर्सवर अवलंबून कार्य करण्याची पद्धत बदलू शकते.
- फंक्शन हे घोषणात्मक भाग, एक एक्झिक्यूटेबल भाग आणि पर्यायी अपवाद-हँडलिंग भाग यांनी बनलेले असते. घोषणा भागामध्ये व्हेरिएबल्सची घोषणा असते. एक्झिक्यूटेबल भागामध्ये लॉजिकचा समावेश असतो, म्हणजे SQL स्टेटमेंट्स आणि अपवाद हाताळणी भाग रन-टाइम दरम्यान कोणतीही त्रुटी हाताळतो.
- ओरॅकल इंजिन फंक्शन कार्यान्वित करण्यासाठी खालील पायऱ्या पार पाडते:
 1. वापरकर्ता प्रवेश सत्यापित करते,

2. प्रक्रिया किंवा कार्य वैधता सत्यापित करते आणि
3. प्रक्रिया किंवा कार्य अंमलात आणते.

4.6.1 फायदे (Advantages)

फंक्शन्सचे विविध फायदे खाली दिले आहेत:

1. ओरॅकल डाटाबेसमध्ये फंक्शन्स संचयित करण्याची क्षमता आपल्याला मॉड्यूलरिटी, देखभालक्षमता आणि कार्यप्रदर्शन सुधारणेमध्ये उत्कृष्ट उंची गाठण्यास अनुमती देते.
2. विशिष्ट कार्ये पूर्ण करण्यासाठी कार्ये वापरणे इतर मॉड्यूलसची विश्वासाहता सुधारते आणि विकास वेळ कमी करते.
3. फंक्शन्स वापरण्याचे इतर काही फायदे आहेत:
 - (i) सुरक्षा,
 - (ii) कामगिरी,
 - (iii) मेमरी वाटप, आणि
 - (iv) उत्पादकता, सचोटी.

4.6.2 फंक्शन तयार करणे (Creating Function)

- फंक्शनने कॉलरला मूल्य परत करणे आवश्यक आहे. फंक्शन कॉलिंग PL/SQL ब्लॉकला फक्त एक मूल्य परत करू शकते.
- एका प्रक्रियेमध्ये एकाधिक आउट पॅरामीटर्स परिभाषित करून, कॉलरला एकाधिक मूल्ये दिली जाऊ शकतात. आउट व्हेरिएबल हे निसर्गाने वैश्विक असल्याने, त्याचे मूल्य कॉलिंग PL/SQL ब्लॉकसह कोणत्याही PL/SQL कोड ब्लॉकद्वारे उपलब्ध आहे.
- फंक्शन तयार करण्यासाठी सिंटॅक्स आहे:

```
CREATE OR REPLACE FUNCTION functionname(argument IN data type, ..) RETURN
data type {IS, AS}
variable declarations; constant declarations;
BEGIN
pl/sql subprogram body;
EXCEPTION
exception pl/sql block;
END;
```

function-name: फंक्शनचे नाव निर्दिष्ट करते.

[OR REPLACE] विद्यमान कार्य सुधारण्याची परवानगी देतो

argument: प्रक्रिया किंवा कार्यासाठी युक्तिवादाचे नाव आहे. कंस असू शकतात कोणतेही युक्तिवाद उपस्थित नसल्यास वगळले.

IN: Specifies that a value for the argument must be specified when calling the procedure or function.

OUT: Specifies that the procedure passes a value for this argument back to its calling environment after execution.

IN OUT: Specifies that a value for the argument must be specified when calling the procedure and that the procedure passes a value for this argument back to its calling environment after execution. By default it takes IN.

RETURN: क्लॉज निर्दिष्ट करते की आपण फंक्शनमधून परत येणारा डाटा प्रकार.

Data type : आर्ग्युमेंटचा डाटा प्रकार आहे.

4.6.3 फंक्शन अंमलात आणणे आणि हटवणे

4.6.3.1. फंक्शन कार्यान्वित करणे:

- फंक्शन तयार करताना, फंक्शनला काय करायचे आहे याची व्याख्या तुम्ही देता. फंक्शन वापरण्यासाठी, तुम्हाला परिभाषित कार्य करण्यासाठी त्या फंक्शनला कॉल करावा लागेल. जेव्हा एखादा प्रोग्राम फंक्शन कॉल करतो तेव्हा प्रोग्राम कंट्रोल कॉल फंक्शनमध्ये हस्तांतरित केला जातो.
- कॉल फंक्शन परिभाषित कार्य करते आणि जेव्हा त्याचे रिटर्न स्टेटमेंट कार्यान्वित केले जाते किंवा जेव्हा ते शेवटचे शेवटचे स्टेटमेंट गाठले जाते तेव्हा ते प्रोग्राम नियंत्रण मुख्य प्रोग्रामकडे परत करते.
- फंक्शन कॉल करण्यासाठी तुम्हाला फंक्शनच्या नावासह आवश्यक पॅरामीटर्स पास करावे लागतील आणि जर फंक्शनने व्हॅल्यू दिली तर तुम्ही रिटर्न केलेली व्हॅल्यू स्टोअर करू शकता.

उदाहरणे:

1. वितर्क वापरून फंक्शनसाठी प्रोग्राम.

```
CREATE OR REPLACE FUNCTION RMT(X IN NUMBER) RETURN NUMBER
IS
BEGIN
  dbms_output.put_line(x*x);
  --return (x*x);
end;
Calling a function:
begin
  dbms_output.put_line(rmt(3));
end;
```

2. deptno स्वीकारणे आणि ते कार्य करण्यासाठी पास करणे आणि विभाग टेबलमध्ये उपस्थित आहे की नाही हे तपासण्याचा कार्यक्रम. जर ते उपस्थित असेल तर प्रिंट क्र. त्या विभागात काम करणाऱ्या कर्मचाऱ्यांची.

```
create or replace function func1(f_no in number) return number,
as
v_no number;
v_count number;
Begin
Select DeptNo into v_no from DEPT where DeptNo=f_no;
If(v_no=f_no) then
Select count(*) into v_count
From EMP
Where DeptNo=f_no;
Return v_count;
End if;
Exception
When NO_DATA_FOUND then
Return -1;
End func1;
/
Calling program:
Declare
```



```

V_no number;
R number;
Begin
V_no:=&v_no;
R:=func1(v_no);
If(R>0) then
dbms_output.put_line('Number of Employees working in
dept'||v_no||'are'||R);
Else
If(R=0) then
dbms_output.put_line('Dept no'||v_no||'exist but no employee working in this dept');
Else
dbms_output.put_line('Dept no'||v_no||' does not exist ');
End if;
End;
/
SQL>/
Enter value for v_no:20
Old 5:v_no:=&v_no;
New 5:v_no:=20;
Number of Employees working in dept 20 are 2

```

3. जास्तीत जास्त दोन मूल्यांची गणना आणि परतावा देणारे कार्य.

```

DECLARE
a number;
b number;
c number;
FUNCTION findMax(x IN number, y IN number)
RETURN number
IS
z number;
BEGIN
IF x > y THEN
z:= x;
ELSE
Z:= y;
END IF;
RETURN z;
END;
BEGIN
a:= 23;
b:= 45;
c := findMax(a, b);
dbms_output.put_line(' Maximum of (23,45): ' || c);
END;
/

```

Maximum of (23,45): 78

PL/SQL procedure successfully completed.

4.6.3.2. फंक्शन हटवणे (Deleting Function):

फंक्शन ड्रॉप करण्यासाठी आम्ही DROP FUNCTION कमांड वापरतो.

वाक्यरचना

DROP FUNCTION functionname;

उदाहरण:

DROP FUNCTION findMax;

4.7 डाटाबेस ट्रिगर (Database Trigger):

- डाटाबेस ट्रिगर हा डाटाबेस टेबल, व्ह्यू किंवा इव्हेंटशी संबंधित एक संग्रहित उपप्रोग्राम आहे. उदाहरणार्थ, टेबलवर INSERT, UPDATE किंवा DELETE स्टेटमेंटचा परिणाम होण्यापूर्वी किंवा नंतर तुम्ही Oracle ला आपोआप ट्रिगर करू शकता.
- ट्रिगर ही एक संग्रहित प्रक्रिया आहे ज्याला ओरॅकल इंजिनद्वारे जेव्हाही, इन्सर्ट, अपडेट किंवा डिलीट स्टेटमेंट फायर केले जाते तेव्हा ते म्हणतात. ट्रिगर ही प्रक्रिया आहे जी डाटाबेसमध्ये संग्रहित केली जाते आणि जेव्हा डाटाबेसमध्ये डीएमएल इव्हेंट घडते तेव्हा ती अस्पष्टपणे चालविली जाते किंवा निश्चित केली जाते.

4.7.1 डाटाबेस ट्रिगर्सचा वापर (Use of Database Trigger)

- डाटाबेस ट्रिगर वापरणे:

1. डाटा स्वतः तयार केला जातो.
2. प्रतिकृती टेबल राखता येते.
3. जटिल अखंडतेची मर्यादा लागू करणे.
4. डाटा बदल संपादित करण्यासाठी.
5. फील्ड स्वयं वाढवण्यासाठी.

4.7.2 ट्रिगरचे प्रकार (Types of Trigger)

- कोणत्या स्तरावर ट्रिगर केले जाते यावर आधारित दोन प्रकारचे ट्रिगर आहेत:

1. रो लेव्हल ट्रिगर (Row Level Trigger):

अपडेट केलेल्या, घातलेल्या किंवा हटवलेल्या प्रत्येक पंक्तीसाठी इव्हेंट ट्रिगर केला जातो. प्रत्येक वेळी टेबलवर ट्रिगरिंग स्टेटमेंटचा परिणाम होतो तेव्हा एक रो ट्रिगर फायर केला जातो. उदाहरणार्थ, अपडेट स्टेटमेंट टेबलच्या अनेक पंक्ती अपडेट करत असल्यास, अपडेट स्टेटमेंटने प्रभावित झालेल्या प्रत्येक पंक्तीसाठी रो ट्रिगर एकदा फायर केला जातो. ट्रिगरिंग स्टेटमेंट कोणत्याही पंक्तीवर परिणाम करत नसल्यास, पंक्ती ट्रिगर चालवला जात नाही. ट्रिगर क्रियेतील कोड ट्रिगरिंग स्टेटमेंटद्वारे प्रदान केलेल्या डाटावर किंवा प्रभावित झालेल्या पंक्तीवर अवलंबून असल्यास रो ट्रिगर उपयुक्त आहेत.

2. स्टेटमेंट लेव्हल ट्रिगर (Statement Level Trigger):

कार्यान्वित केलेल्या प्रत्येक sql स्टेटमेंटसाठी इव्हेंट ट्रिगर केला जातो. कोणत्याही पंक्तीवर परिणाम होत नसला तरीही, ट्रिगरिंग स्टेटमेंट टेबलमधील पंक्तींच्या संख्येकडे दुर्लक्ष करून, ट्रिगरिंग स्टेटमेंटच्या वतीने एकदा स्टेटमेंट ट्रिगर केला जातो. उदाहरणार्थ, DELETE स्टेटमेंट टेबलमधून अनेक पंक्ती हटवल्यास, स्टेटमेंट-स्तरीय DELETE ट्रिगर फक्त एकदाच फायर केला जातो. ट्रिगर क्रियेतील कोड ट्रिगरिंग स्टेटमेंटने प्रदान केलेल्या डाटावर किंवा प्रभावित झालेल्या पंक्तीवर अवलंबून नसल्यास स्टेटमेंट ट्रिगर उपयुक्त आहेत. उदाहरणार्थ, यासाठी स्टेटमेंट ट्रिगर वापरा:

- वर्तमान वेळ किंवा वापरकर्त्यावर एक जटिल सुरक्षा तपासणी करा
- एकच ऑडिट रेकॉर्ड व्युत्पन्न करा.

4.7.3 ट्रिगर तयार करण्यासाठी वाक्यरचना (Creating Trigger)

ट्रिगर तयार करण्यासाठी वाक्यरचना खाली दिली आहे:

```
CREATE [OR REPLACE ] TRIGGER triggername
    [before/after/instead of]
    [insert/update/delete]
    on tablename [for each statement/ for each row][when condition]
    PL/SQL block.
```

- TRIGGER trigger_name तयार करा [किंवा बदला] - दिलेल्या नावाने ट्रिगर तयार करते किंवा त्याच नावाने विद्यमान ट्रिगर ओव्हरराइट करते.
- [पूर्वी / नंतर / ऐवजी] - ट्रिगर कोणत्या वेळी फायर झाला पाहिजे हे सूचित करते. उदाहरणार्थ: टेबल अपडेट करण्यापूर्वी किंवा नंतर. ऐवजी दृश्यावर ट्रिगर तयार करण्यासाठी वापरला जातो. दृश्यावर ट्रिगर तयार करण्यासाठी आधी आणि नंतर वापरले जाऊ शकत नाही.
- [इन्सर्ट / अपडेट / डिलीट] - ट्रिगरिंग इव्हेंट निर्धारित करते. एकापेक्षा जास्त ट्रिगरिंग इव्हेंट्स OR कीवर्डद्वारे विभक्त करून एकत्र वापरले जाऊ शकतात. सर्व निर्दिष्ट ट्रिगरिंग इव्हेंटवर ट्रिगर फायर केला जातो.
 1. इन्सर्ट कमांड : घातल्या जाणाऱ्या फील्डचे मूल्य :new च्या आधी असणे आवश्यक आहे.
 2. अद्ययावत आदेश : मूळ मूल्य :old ने ऍक्सेस केले जाते आणि नवीन मूल्ये :new च्या आधी असतील.
 3. डिलीट कमांड : या केसमधील मूल्ये :old च्या आधी असणे आवश्यक आहे.
- [प्रत्येक पंक्तीसाठी] - प्रत्येक पंक्ती प्रभावित झाल्यावर ट्रिगर फायर झाला पाहिजे की नाही हे निर्धारित करण्यासाठी वापरले जाते (म्हणजेच एक पंक्ती स्तर ट्रिगर) किंवा संपूर्ण SQL स्टेटमेंट कार्यान्वित झाल्यावर (म्हणजे स्टेटमेंट लेव्हल ट्रिगर) फक्त एकदाच.
- WHEN (अट) - हे कलम फक्त पंक्ती स्तर ट्रिगरसाठी वैध आहे. ट्रिगर फक्त निर्दिष्ट केलेल्या अटी पूर्ण करणाऱ्या पंक्तीसाठी फायर केला जातो.

उदाहरणार्थ :

```
create or replace trigger trg1
before insert on EMP
for each row
begin
if :new.Sal<=0 then
raise_application_error('salary should be greater than 0');
end if
end;
SQL>/
Trigger created
```

ट्रिगर सक्षम/अक्षम करणे:

विशिष्ट ट्रिगर सक्षम किंवा अक्षम करण्यासाठी ALTER TRIGGER कमांड वापरली जाते.

वाक्यरचना:

```
Alter trigger trigger_name Enable/Disable;
```

ट्रिगर तयार केल्यावर, तो आपोआप सक्षम होतो. आम्ही वापरत असलेल्या EMP सारणीचे सर्व ट्रिगर अक्षम करण्यासाठी. सारणी बदला EMP सर्व ट्रिगर अक्षम करा;

4.7.4 ट्रिगर हटवत आहे (Deleting Trigger)

ट्रिगर हटवण्यासाठी आम्ही DROP TRIGGER कमांड वापरतो.

वाक्यरचना:

Drop trigger trigger_name;

उदाहरण

DROP TRIGGER trg1;

References:

1. <https://nptel.ac.in/courses/106105175>
2. <https://www.w3schools.com/sql/>
3. <https://www.tutorialspoint.com/sql/index.htm>
4. Database System Concepts by Henry F. Korth McGraw Hill Education

युनिट - 5

डाटाबेस ऍडमिनिस्ट्रेशन (Database Administration)

विषय निष्पत्ती (Course Outcome): डाटाबेसवर सुरक्षा आणि बॅकअप पद्धती लागू करणे

घटक निष्पत्ती (Theory Learning Outcome):

1. डाटाबेस प्रशासनासाठी SQL केरी लागू करणे.
2. विविध प्रकारच्या डाटाबेस बॅकअप प्रक्रियेची संकल्पना स्पष्ट करणे.
3. प्रगत डाटाबेस संकल्पनांशी संबंधित विविध संज्ञांचे वर्णन करणे.

5. 1 डाटाबेस प्रशासनाचा परिचय (Introduction to database administration):

डाटाबेस व्यवस्थापन प्रणालीच्या (DBMS) कार्यामध्ये एकरूपता(concurrency), सुरक्षा(security), बॅकअप(backup) आणि पुनर्प्राप्ती(recovery), अखंडता(Integrity) आणि डाटाचे वर्णन(data description) समाविष्ट आहे. डाटाबेस मॅनेजमेंट सिस्टीम (DBMS) अनेक महत्त्वाचे फायदे प्रदान करतात परंतु अंमलबजावणी करणे महाग असू शकते आणि यात बराच वेळ जाऊ शकतो.

5. 1. 1. डाटाबेस वापरकर्त्यांचे प्रकार (Types of Database Users):

1. डाटाबेस प्रशासक (DBA): डाटाबेस ऍडमिनिस्ट्रेटर (DBA) ही एक व/संघ आहे, जी स्कीमा परिभाषित करते आणि डाटाबेसच्या 3 स्तरांवर देखील नियंत्रण ठेवते. DBA नंतर वापरकर्त्यासाठी नवीन खाते आयडी आणि पासवर्ड तयार करेल, जर त्याला/तिला डाटाबेसमध्ये प्रवेश करण्याची आवश्यकता असेल. डाटाबेसला सुरक्षा प्रदान करण्यासाठी देखील DBA जबाबदार आहे आणि तो केवळ अधिकृत वापरकर्त्यांना डाटाबेसमध्ये प्रवेश/सुधारित करण्याची परवानगी देतो. सुरक्षा भंग आणि खराब प्रणाली प्रतिसाद वेळ यासारख्या समस्यांसाठी DBA जबाबदार आहे.

2. डीबीएमएस (DBMS) चे ज्ञान नसणारे वापरकर्ते (Naive Users): वापरकर्ते हे अप्रत्यक्ष वापरकर्ते आहेत, ज्यांना डाटाबेस व्यवस्थापन प्रणालीचे (DBMS) कोणतेही ज्ञान नाही परंतु ते इच्छित परिणाम मिळविण्यासाठी त्यांच्या दैनंदिन जीवनात डाटाबेस अनुप्रयोगांचा वारंवार वापर करतात. उदाहरणार्थ, रेल्वेचे तिकीट बुकिंग करणारे हे भोळे वापरकर्ते आहेत. कोणत्याही बँकेतील लिपिक हे एक डीबीएमएस (DBMS) चे ज्ञान नसणारे वापरकर्ते असतात कारण त्यांना डाटाबेस व्यवस्थापन प्रणालीचे (DBMS) ज्ञान नसते पण तरीही ते डाटाबेस वापरतात आणि दिलेले काम करतात.

3. सिस्टम/प्रणाली विश्लेषक (System Analyst): सिस्टम विश्लेषक एक वापरकर्ता आहे, जो अंतिम वापरकर्त्यांच्या आवश्यकतांचे विश्लेषण करतो. अंतिम वापरकर्त्यांच्या(end user) सर्व गरजा पूर्ण झाल्या आहेत का ते, ते तपासतात.

4. अत्याधुनिक/ज्ञानी वापरकर्ते (Sophisticated Users): अत्याधुनिक वापरकर्ते अभियंते, शास्त्रज्ञ, व्यवसाय विश्लेषक असू शकतात, जे डाटाबेसशी परिचित आहेत. ते त्यांच्या गरजेनुसार स्वतःचे डाटाबेस ऍप्लिकेशन विकसित करू शकतात. ते प्रोग्राम कोड लिहित नाहीत परंतु ते थेट केरी प्रोसेसरद्वारे SQL केरी लिहून डाटाबेसशी संवाद साधतात.

5. ऍप्लिकेशन प्रोग्रामर (Application Programmer): ऍप्लिकेशन प्रोग्रामर(Application Programmer) ज्यांना सॉफ्टवेअर अभियंता असेही संबोधले जाते, ते बॅक-एंड प्रोग्रामर आहेत जे ऍप्लिकेशन प्रोग्रामसाठी कोड लिहितात. ते संगणक व्यावसायिक आहेत. हे प्रोग्राम Visual Basic, Developer, C, FORTRAN, COBOL इत्यादी प्रोग्रामिंग भाषांमध्ये लिहिले जाऊ शकतात.

6. विशेष वापरकर्ते: विशेष वापरकर्ते हे अत्याधुनिक वापरकर्ते आहेत जे विशिष्ट डाटाबेस अनुप्रयोग लिहितात जे पारंपारिक डाटा-प्रोसेसिंग फ्रेमवर्कमध्ये बसत नाहीत. या ऍप्लिकेशन्समध्ये कॉम्प्युटर एडेड-डिझाइन सिस्टीम, नॉलेज बेस आणि एक्स्पर्ट सिस्टीम इ. चा सामावेश होतो.

5.1.2. वापरकर्ते तयार करणे आणि हटवणे (Create and Delete Users) :

SQL मधील 'CREATE USER' ही कमांड SQL डाटाबेसमध्ये नवीन वापरकर्ता तयार करण्यासाठी वापरली जाणारी कमांड आहे. ही कमांड डाटा कंट्रोल लॅंग्वेज (Data Control Language DCL) चा एक भाग आहे, ज्यामध्ये वापरकर्ता परवानग्या नियंत्रित करणाऱ्या 'GRANT' आणि 'REVOKE' सारख्या आदेशांचा समावेश आहे. नवीन वापरकर्ता तयार करताना, अनेकदा पासवर्ड निर्दिष्ट करणे आवश्यक असते. हे 'CREATE USER' विधानासह 'IDENTIFIED BY' क्लॉज वापरून केले जाऊ शकते.

Syntax (मांडणी):

```
CREATE USER user_name IDENTIFIED BY password;
```

Example (उदाहरण):

```
CREATE USER Mayur IDENTIFIED BY guest@123;
```

येथे युजर(वापरकर्ता) मयूर आहे आणि त्याचा पासवर्ड guest@123 आहे.

5.1.3. वापरकर्त्यांना विशेषाधिकार नियुक्त करणे (Assign privileges to users)

डाटाबेसला वापरकर्त्यांच्या परवानग्या देण्यासाठी 'GRANT' विधान वापरले जाते. नवीन वापरकर्ता (Create user) तयार केल्यानंतर, तुम्हाला सहसा त्यांना SELECT, INSERT, UPDATE, DELETE इत्यादी विविध कमांड अंमलात आणण्यासाठी परवानग्या द्याव्या लागतात. हे 'GRANT' कमांड वापरून साध्य करता येते.

Syntax (मांडणी):

```
GRANT privilege_name
ON object_name
TO {user_name |PUBLIC |role_name}
[WITH GRANT OPTION];
```

Example (उदाहरण):

```
GRANT SELECT, INSERT, DELETE ON emp TO Mayur ;
```

येथे मयूर SELECT, INSERT, DELETE या कमांड EMP या टेबलवर अंमलात आणू शकतो.

- GRANT ALL स्टेटमेंटचा वापर वापरकर्त्याला सर्व परवानग्या देण्यासाठी किंवा टेबल, व्ह्यू किंवा डाटाबेससारख्या विशिष्ट डाटाबेस ऑब्जेक्टसाठी भूमिका देण्यासाठी केला जातो. हे विधान विशेषतः डाटाबेस प्रशासकांसाठी उपयुक्त आहे, ज्यांना डाटाबेस हाताळण्याच्या सर्व कमांड्स वापरकर्त्यांना द्यायच्या आहेत.

Syntax (मांडणी):

```
GRANT ALL PRIVILEGES ON [object] TO [user/role];
```

Example (उदाहरण):

```
GRANT ALL PRIVILEGES ON EMP TO Mayur ;
```

येथे मयूर या युजरला EMP या टेबलवर सर्व कमांड्स अंमलात आणण्याच्या परवानगी देण्यात आल्या आहेत.

- 'REVOKE' विधान वापरकर्त्याकडून परवानग्या परत काढून घेण्यासाठी वापरले जाते.

Syntax (मांडणी):

```
REVOKE privilege_name
```

ON object_name

FROM {user_name |PUBLIC |role_name}

Example (उदाहरण):

REVOKE INSERT ON emp FROM 'user1'@'localhost';

येथे मयूरकडून EMP टेबलवर असलेली INSERT या कमांडची परवानगी काढून घेण्यात आली आहे. आता तो EMP टेबलवर INSERT ही कमांड अंमलात आणू शकत नाही.

- REVOKE ALL स्टेटमेंटचा वापर वापरकर्त्याच्या टेबल, व्ह्यू, किंवा डाटाबेसच्या सर्व परवानग्या काढून घेण्यासाठी वापर केला जातो. हे विधान विशेषतः डाटाबेस प्रशासकांसाठी उपयुक्त आहे, ज्यांना डाटाबेस हाताळण्याच्या सर्व कमांड्स कडून काढून घ्यायच्या आहेत.

Syntax (मांडणी):

REVOKE ALL PRIVILEGES ON [object] FROM [user/role];

Example (उदाहरण):

REVOKE ALL PRIVILEGES ON TABLE emp FROM Mayur;

येथे EMP टेबलच्या असलेल्या सर्व कमांडच्या परवानगी मयूर कडून काढून घेण्यात आल्या आहेत

5.2 व्यवहार (Transaction):

5.2.1. संकल्पना: डाटाबेस व्यवस्थापन प्रणालीमध्ये (DBMS) व्यवहार (Transaction) म्हणजे एक किंवा अधिक SQL ऑपरेशन्स (जसे की insert, update, delete, किंवा select) एकाच तर्कसंगत कामाचे एकक म्हणून केलेली क्रिया आहे. या ऑपरेशन्समध्ये सर्व यशस्वीरीत्या पूर्ण होणे आवश्यक आहे किंवा एकही पूर्ण होऊ नये, यामुळे डाटाबेस सुसंगत स्थितीत राहतो. व्यवहार डाटाबेसची अखंडता आणि विश्वासार्हता राखण्यासाठी अत्यंत महत्वाचे आहेत, विशेषतः ज्या वातावरणात, अनेक वापरकर्ते किंवा अनुप्रयोग डाटाबेसमध्ये प्रवेश करतात आणि हेरफेर करतात.

व्यवहार काही अवस्थांमधून जातो. या अवस्था, व्यवहाराच्या सद्यस्थितीबद्दल सांगतात आणि आम्ही पुढे व्यवहारात प्रक्रिया कशी करू हे देखील सांगतात. या अवस्था नियमांचे संचालन करतात, जे व्यवहाराचे भवितव्य ठरवतात की तो करायचा की रद्द करायचा की पुढे सुरु ठेवायचा.

5.2. 2. व्यवहाराच्या अवस्था (States of Transaction):

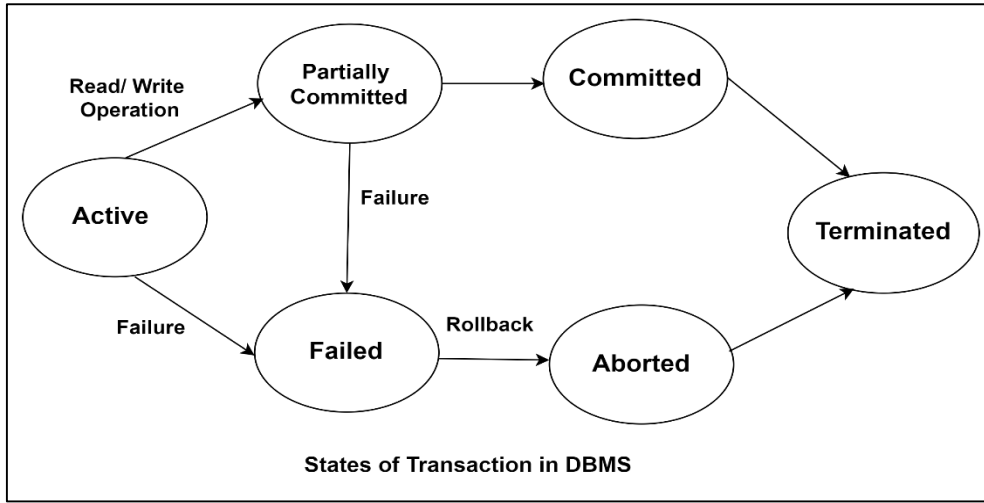
ACID गुण (ACID properties) ह्या डाटाबेस व्यवस्थापन प्रणालीतील महत्वाच्या गुणांचा समूह आहे ज्यांनी व्यवहाराची निष्पत्ती आणि स्थिती सुरक्षितता सुनिश्चित करतात. खासगी डाटाबेस व्यवस्थापन प्रणाल्यात, बँकअप आणि पुनर्प्राप्ती, सामान्यतः ACID गुण अनुसरले जातात. या गुणांची मुख्य प्रकारे आहेत:

- **Atomicity (अखंडता):** या गुणाचे उपयोग केल्यास, व्यवहार पूर्णत्वाने किंवा न किंवा एकही वेळेस पूर्ण केले जाते. जर कोणत्याही घटनेनंतर त्रुटी आली तर व्यवहार रद्द होतो आणि डाटाबेस अवशिष्ट स्थितीत राहतो.
- **Consistency (संचितता):** या गुणाचे उपयोग केल्यास, व्यवहारानंतर डाटाबेस संचित असावा. व्यवहारानंतर डाटाबेस संपूर्णपणे संचित आहे आणि कोणत्याही संदर्भात असुसंगतता नाही असे सांगता येऊ शकते.
- **Isolation (पृथक्ता):** या गुणाचे उपयोग केल्यास, एक व्यवहार इतर व्यवहारांपासून पूर्णपणे पृथक् आहे आणि व्यवहारांतील कोणत्याही परस्परबंधांची असुसंगतता नाही असे सांगता येऊ शकते.
- **Durability (स्थिरता):** या गुणाचे उपयोग केल्यास, व्यवहार पूर्ण झाल्यानंतर त्याचे परिणाम डाटाबेसमध्ये स्थायी राहतात आणि कोणत्याही हानीची संभावना नाही, जसे की प्रणाली संक्रांती किंवा संघटना.

या ACID गुणांचा उपयोग करून, व्यवहारांची निष्पत्ती आणि स्थिती सुरक्षितता सुनिश्चित केली जाते, ज्याने डाटाबेसमध्ये सामान्यतः संचितता आणि स्थिरता सुरक्षित करते.

5.2.3. व्यवहाराच्या अवस्था (States of Transaction):

वेगवेगळ्या प्रकारच्या व्यवहाराच्या अवस्थांचा समावेश आहे:



आकृती 5.1 डाटाबेस व्यवहाराच्या विविध अवस्था (State of Transaction of DBMS)

सक्रिय अवस्था (Active State): जेव्हा व्यवहाराच्या सूचनांची अंमलबजावणी होत असते, तेव्हा व्यवहार सक्रिय अवस्थेत असतो. सर्व 'वाचण्याच्या आणि लिहिण्याच्या' ऑपरेशन्स कोणत्याही त्रुटीशिवाय पूर्ण झाल्यास, तो "आंशिकपणे पूर्ण केलेली अवस्था" मध्ये जातो; कोणतीही सूचना अयशस्वी झाल्यास, तो "अयशस्वी अवस्था" मध्ये जातो.

आंशिकपणे पूर्ण अवस्था (Partially Committed): सर्व वाचन आणि लेखन ऑपरेशन्स पूर्ण झाल्यानंतर, बदल मुख्य स्मृती किंवा स्थानिक बफरमध्ये केले जातात. जर बदल डाटाबेसवर कायमचे केले गेले तर अवस्था "पूर्ण केली अवस्था" मध्ये बदलते आणि अपयश झाल्यास तो "अयशस्वी अवस्था" मध्ये जातो.

अयशस्वी अवस्था (Failed State): व्यवहाराची कोणतीही सूचना अयशस्वी झाल्यास किंवा डाटाबेसवर कायम बदल करण्यामध्ये अपयश आल्यास, तो "अयशस्वी अवस्था" मध्ये जातो.

रद्द केलेली अवस्था (Aborted State): कोणत्याही प्रकारच्या अपयशानंतर व्यवहार "अयशस्वी अवस्था" मधून "रद्द केलेली अवस्था" मध्ये जातो आणि कारण मागील अवस्थांमध्ये, बदल फक्त स्थानिक बफर किंवा मुख्य स्मृतीमध्ये केले जातात, त्यामुळे हे बदल हटवले जातात किंवा रोल-बॅक केले जातात.

पूर्ण केली अवस्था (Committed State): ही अवस्था तेव्हा असते जेव्हा डाटाबेसवर बदल कायम केले जातात आणि व्यवहार पूर्ण होतो आणि "समाप्त अवस्था" मध्ये संपुष्टात येतो.

समाप्त अवस्था (Terminated State): जर कोणतेही रोल-बॅक नसेल किंवा व्यवहार "पूर्ण केली अवस्था" मधून आला असेल, तर प्रणाली सुसंगत असते आणि नवीन व्यवहारासाठी तयार असते आणि जुना व्यवहार समाप्त केला जातो.

5.3. डाटाबेस बॅकअप(Database Backup):

5.3.1. डाटाबेस बॅकअप(Database Backup)

बॅकअप ही डाटाची प्रत आहे आणि इतरत्र संग्रहित केली जाते जेणेकरून डाटा गमावण्याच्या घटनेनंतर मूळ पुनर्संचयित करण्यासाठी त्याचा वापर केला जाऊ शकतो.

डाटाबेस व्यवस्थापन प्रणालीत (DBMS) बिघाडांसाठी (फेल्योर्सचे) विविध कारणे असू शकतात, ज्यामुळे डाटाबेस व्यवस्थापन प्रणालीच्या सामान्य कार्यपद्धतीत बाधा उत्पन्न होतात.

5.3.2 बिघाडांचे प्रकार (Types of Failure)

खासगी DBMS मध्ये बिघाडांचे (फेल्योर्सचे) काही सामान्य प्रकार आहेत: डाटाबेस मॅनेजमेंट सिस्टम (DBMS) मध्ये, विविध कारणांमुळे बिघाड होऊ शकतो, ज्यामुळे सिस्टमच्या सामान्य ऑपरेशनमध्ये व्यत्यय येतो. DBMS मधील काही सामान्य प्रकारचे बिघाड येथे आहेत:

- **हार्डवेअर बिघाड (Hardware Failure):** भौतिक घटक जसे की डिस्क ड्राइव्ह, मेमरी मॉड्यूल किंवा नेटवर्क डिवाइसेसच्या अपयशामुळे डाटा गमावणे किंवा सिस्टम डाउनटाइम होऊ शकतो.
- **सॉफ्टवेअर बिघाड (Software Failure):** डीबीएमएस सॉफ्टवेअरमधील दोष, त्रुटी किंवा त्रुटींमुळे सिस्टम क्रॅश किंवा अनपेक्षित वर्तन होऊ शकते, ज्यामुळे डाटा विसंगतता किंवा तोटा होऊ शकतो.
- **नेटवर्क बिघाड (Network Failure):** नेटवर्क कनेक्टिव्हिटीच्या समस्या किंवा क्लायंट आणि सर्व्हर सिस्टममधील संप्रेषण अपयश डाटाबेस ऑपरेशन्समध्ये व्यत्यय आणू शकतात, ज्यामुळे व्यवहार अयशस्वी होऊ शकतात किंवा डाटा अॅक्सेसेबल होऊ शकतो.
- **वीज बिघाड किंवा वीज खंड (Electric Failure):** वीज(पॉवर) आउटेज किंवा अचानक वीज गळतीमुळे DBMS सिस्टीम अचानक बंद होऊ शकते, योग्य बॅकअप यंत्रणा नसल्यास डाटा करप्ट किंवा नुकसान होऊ शकते.
- **व्यवहार खंड (Transaction Failure):** डाटा प्रोसेसिंगमधील त्रुटींमुळे, अखंडतेच्या मर्यादांचे उल्लंघन किंवा संसाधन विवाद समस्यांमुळे वैयक्तिक व्यवहारांमध्ये होणाऱ्या अपयशांमुळे व्यवहार रोलबॅक किंवा संपुष्टात येऊ शकतो.
- **मीडिया बिघाड (Media Failure):** स्टोरेज मीडिया दूषित किंवा भौतिक नुकसान झाल्यामुळे डाटा गमावणे किंवा भ्रष्टाचार होऊ शकतो, ज्यामुळे डाटाबेसची उपलब्धता आणि विश्वासार्हता प्रभावित होते.
- **मानवी चूक (Human Error):** योग्य खबरदारी आणि प्रवेश नियंत्रणे न घेतल्यास अपघाती हटवणे, बदल करणे किंवा वापरकर्ते किंवा प्रशासकांद्वारे डाटाच्या चुकीच्या हाताळणीमुळे डाटा विसंगती किंवा तोटा होऊ शकतो.
- **नैसर्गिक आपत्ती (Natural Calamities):** भूकंप, पूर, आग किंवा इतर नैसर्गिक आपत्ती यासारख्या घटनांमुळे डाटा केंद्रे किंवा पायाभूत सुविधांना भौतिकरित्या नुकसान होऊ शकते, ज्यामुळे व्यापक व्यत्यय आणि डाटाचे संभाव्य नुकसान होऊ शकते.
- **सुरक्षा भंग (Security Breach):** अनधिकृत प्रवेश, हॅकिंगचे प्रयत्न किंवा सुरक्षेचे उल्लंघन डाटाबेसची गोपनीयता, अखंडता किंवा उपलब्धतेशी तडजोड करू शकते, ज्यामुळे डाटा चोरी, फेरफार किंवा विनाश होऊ शकतो.
- **समवर्ती नियंत्रण अयशस्वी (Concurrency Control Failure):** बहु-वापरकर्ता वातावरणात, समवर्ती नियंत्रण यंत्रणा सामायिक केलेल्या डाटामध्ये प्रवेश योग्यरित्या समन्वयित करण्यात अयशस्वी होऊ शकते, ज्यामुळे डाटा विसंगती किंवा समवर्ती व्यवहारांमधील संघर्ष होऊ शकतो.

या प्रकारच्या बिघाडांमुळे मजबूत बॅकअप आणि पुनर्प्राप्ती धोरणे (रिकव्हरी स्ट्रॅटेजीज) अंमलात आणण्याचे महत्त्व अधोरेखित होते, तसेच बिघाडाचा प्रभाव कमी करण्यासाठी आणि डाटाबेस सिस्टमची विश्वासार्हता आणि उपलब्धता सुनिश्चित करण्यासाठी रिडंडंसी(डाटाच्या अनावश्यक प्रती), फॉल्ट टॉलरन्स(फॉल्ट टॉलरन्स ही एक अशी प्रक्रिया आहे जी ऑपरेटिंग सिस्टमला हार्डवेअर किंवा सॉफ्टवेअरमधील अपयशाला प्रतिसाद देण्यास सक्षम करते) आणि आपत्ती पुनर्प्राप्ती यासारख्या यंत्रणा वापरल्या जातात.

5.3.3. डाटाबेस व्यवस्थापन प्रणालीतील (DBMS) बिघाड(Causes of Failure) :

डाटाबेस व्यवस्थापन प्रणालीतील (DBMS) बिघाड होण्याची कारणे विविध असू शकतात. इथे डाटाबेस बिघाड होण्याची प्रमुख कारणांची सविस्तर माहिती दिली आहे:

1. हार्डवेअर बिघाड (Hardware Failure):

हार्डवेअर बिघाड होणे डाटाबेस व्यवस्थापन प्रणालीच्या सर्वात सामान्य आणि गंभीर समस्यांपैकी एक आहे. यामध्ये खालील समस्यांचा समावेश होतो:

- **डिस्क ड्राइव्ह फेल्युअर:** हार्ड डिस्क, एसएसडी किंवा अन्य स्टोरेज डिव्हायसेसचे अपयश झाल्यास डाटाची हानी होऊ शकते.
- **मेमरी मॉड्यूल फेल्युअर:** रॅम किंवा अन्य मेमरी घटकांचे अपयश झाल्यास डाटाबेस प्रणाली ठप्प होऊ शकते.
- **नेटवर्क उपकरण फेल्युअर:** नेटवर्क राऊटर, स्विच किंवा अन्य नेटवर्क घटकांचे अपयश झाल्यास डाटाबेस सेवांमध्ये अडथळा येऊ शकतो.

2. सॉफ्टवेअर बिघाड (Software Failure):

सॉफ्टवेअरच्या समस्यांमुळे डाटाबेस प्रणाली बिघडू शकते. यामध्ये खालील समस्या असू शकतात:

- **सॉफ्टवेअर बग्स:** डीबीएमएस सॉफ्टवेअरमधील बग्स किंवा त्रुटीमुळे प्रणाली अनपेक्षितपणे बंद होऊ शकते किंवा चुकीचे वर्तन करू शकते.
- **सिस्टम क्रॅश:** डीबीएमएस किंवा त्यावर चालणारे ऑपरेटिंग सिस्टम क्रॅश झाल्यास डाटाबेस प्रणाली ठप्प होऊ शकते.

3. नेटवर्क बिघाड (Network Failure):

नेटवर्क बिघाड झाल्यास डाटाबेसमध्ये प्रवेश आणि डाटा ट्रान्सफरमध्ये अडथळा येऊ शकतो. यामध्ये खालील समस्या येऊ शकतात:

- **नेटवर्क कनेक्टिव्हिटी समस्या:** नेटवर्क कनेक्शनमध्ये अडथळे किंवा ब्रेकडाउनमुळे डाटाबेसमध्ये प्रवेश करता येत नाही.
- **नेटवर्क लेटन्सी:** उच्च नेटवर्क लेटन्सीमुळे डाटाबेस व्यवहार (ट्रान्सॅक्शन)मध्ये विलंब होऊ शकतो.

4. वीज अनुत्तीर्ण:

वीजपुरवठ्यातील अडथळे किंवा अचानक वीज जाणे यामुळे डाटाबेस प्रणाली अनपेक्षितपणे बंद होऊ शकते:

- **वीजपुरवठा खंडित होणे:** अचानक वीजपुरवठा खंडित झाल्यास डाटाबेस प्रणाली बंद होऊ शकते, ज्यामुळे डाटा हानी होऊ शकते.
- **वीजपुरवठा शॉर्टेज:** कमी वीजपुरवठा किंवा अनियमित वीजपुरवठ्यामुळे प्रणाली स्थिरतेत अडथळा येऊ शकतो.

5. व्यवहार बिघाड :

व्यवहाराच्या अंमलबजावणीत त्रुटी आल्यास त्यात बिघाड होऊ शकतो:

- **डाटा प्रक्रिया त्रुटी:** वाचन आणि लेखन प्रक्रियेत त्रुटी आल्यास व्यवहार अपयशी होतो.
- **संसाधन संघर्ष:** अनेक व्यवहार एकाच डाटा स्रोतावर कार्य करीत असल्यास संसाधन संघर्ष होऊ शकतो, ज्यामुळे व्यवहार अनुत्तीर्ण होऊ शकतो.

6. मीडिया बिघाड (Media Failure):

संचित मिडियामधील समस्यांमुळे डाटाची हानी होऊ शकते:

- **डिस्क करप्शन:** हार्ड डिस्क किंवा अन्य स्टोरेज मिडियामधील डाटा करप्शनमुळे डाटाची हानी होऊ शकते.
- **फिजिकल डॅमेज:** स्टोरेज मिडियाचे नुकसान झाल्यास डाटाची हानी होऊ शकते.

7. मानवी त्रुटी (Human Mistakes):

वापरकर्त्यांच्या चुका किंवा अनियमित क्रियांमुळे डाटाबेसमध्ये बिघाड होऊ शकते:

- **आकस्मिक डिलीशन:** वापरकर्त्याद्वारे आकस्मिकपणे डाटा डिलीट केल्यास डाटाची हानी होऊ शकते.
- **चुकीचे अपडेट्स:** वापरकर्त्याद्वारे चुकीच्या डाटाची अद्यतने(Update बदल) केल्यास डाटाबेस असंगत होऊ शकते.

8. नैसर्गिक आपत्ती (Natural Calamity):

नैसर्गिक आपत्तींमुळे डाटाबेसची शारीरिक सुरक्षितता धोक्यात येऊ शकते:

- **भूकंप:** भूकंपामुळे डाटा सेंटरचे नुकसान होऊ शकते.
- **पूर:** पूरामुळे डाटाबेसच्या सर्व्हरचे नुकसान होऊ शकते.

9. सुरक्षा उल्लंघन:

अनधिकृत प्रवेश किंवा हॅकिंग प्रयत्नांमुळे डाटाबेस धोक्यात येऊ शकतो:

- **डाटा चोरी:** अनधिकृत वापरकर्ते डाटा चोरी करू शकतात.
- **डाटा बदल:** हॅकर्स डाटामध्ये अनधिकृत बदल करू शकतात.

10. सहकालता नियंत्रण अपयश

सहकालता नियंत्रण अपयश म्हणजे एकाच वेळी अनेक व्यवहार (transactions) डाटाबेसमध्ये कार्यरत असताना त्यांच्यातील समन्वय नीट न होणे. यामुळे डाटाच्या असंगती आणि विसंगतीच्या समस्या निर्माण होऊ शकतात. सहकालता नियंत्रणाचे अनुत्तीर्ण होणे म्हणजे एकाच डाटावर अनेक व्यवहार एकाच वेळी काम करत असताना डाटाच्या अखंडतेत(Integrity) अडथळा निर्माण होणे. बहु-उपयोगकर्ता वातावरणात सहकालता नियंत्रण व्यवस्थित न झाल्यास डाटाची असंगती होऊ शकते:

डाटा संघर्ष: अनेक व्यवहार एकाच डाटा स्रोतावर एकाच वेळी कार्य करीत असल्यास डाटा संघर्ष होऊ शकतो. या बिघाडांच्या कारणांमुळे डाटाबेस प्रणालीच्या विश्वसनीयतेवर परिणाम होतो. त्यामुळे, डाटाबेसच्या सुरक्षिततेसाठी मजबूत बॅकअप आणि पुनर्प्राप्ती योजना तसेच त्रुटी सहनशीलता आणि आपत्ती पुनर्प्राप्ती प्रणालींचा वापर करणे महत्त्वाचे आहे.

5.3.4. डाटाबेस बॅकअप संकल्पना (Database Backup Introduction) :

डाटाबेस बॅकअप म्हणजे डाटाबेसमधील सर्व डाटा कॉपी करून सुरक्षित ठिकाणी साठवणे. हे बॅकअप आवश्यक आहेत कारण त्यांमुळे डाटा गमावल्यास किंवा भ्रष्ट झाल्यास पुनर्प्राप्त करणे शक्य होते. बॅकअपमुळे डाटाची अखंडता, सुरक्षा आणि उपलब्धता सुनिश्चित होते. डाटाबेस बॅकअपची दोन मुख्य प्रकारे आहेत: फिजिकल बॅकअप आणि लॉजिकल बॅकअप.

5.3.5. डाटाबेस बॅकअपचे प्रकार (Types of Database Backup) :

डाटाबेस बॅकअपचे प्रकार

1. फिजिकल बॅकअप (Physical Backup)
2. लॉजिकल बॅकअप (Logical Backup)

1. फिजिकल बॅकअप (Physical Backup): फिजिकल बॅकअपमध्ये डाटाबेसच्या फाईल्सची प्रतिमा (Image) किंवा प्रत(कॉपी) घेण्यात येते. हे बॅकअप फाइल सिस्टमवर आधारित असतात आणि डाटाबेसच्या ब्लॉक्स आणि फाईल्सची प्रत (कॉपी) करतात.

फिजिकल बॅकअपचे वैशिष्ट्ये:

- **द्रुत गतीने बॅकअप:** फिजिकल बॅकअप द्रुत गतीने घेतले जाऊ शकतात कारण ते संपूर्ण फाईल्सची कॉपी करतात. हे सामान्यतः मोठ्या डाटाबेससाठी उपयुक्त असते.
- **डाटाची अखंडता:** या बॅकअपमध्ये डाटाची अखंडता टिकवली जाते कारण संपूर्ण डाटाबेसची प्रतिमा घेतली जाते, ज्यामुळे डाटा गमावण्याची शक्यता कमी असते.

- **डाटा पुनर्प्राप्ती (रीस्टोरेशन) सोपे:** फिजिकल बॅकअपसमधून डाटाबेस पुन्हा लवकर पुन्हा प्राप्त करू शकतो, रीस्टोर करता येते. संपूर्ण फाईल्स पुन्हा लोड केल्याने हे रीस्टोरेशन द्रुत होते.

फिजिकल बॅकअपचे फायदे:

- केवळ वापरकर्त्यांना संक्षेपात डाटाबेस पुनर्स्थापित करण्याच्या आवश्यकता असताना हे उपयोगी आहे
- ते डाटाबेसमध्ये केलेल्या व्यवहारांच्या आणि बदलांची माहिती पुरवतात.

फिजिकल बॅकअपचे तोटे :

- डाटाबेस व्यवस्थापन प्रणालीची काम करण्याची गती धीमी होते.

2. लॉजिकल बॅकअप (Logical Backup)

लॉजिकल बॅकअपमध्ये एक डाटाबेसमध्ये असलेली माहिती आहे, ज्याचा स्वरूप डाटाबेसच्या संरचनेशी संबंधित आहे, आणि त्याचा प्रयोग डाटाच्या संरचनेसाठी कामात आलेल्या प्रक्रिया आणि कार्यक्षमतेसाठी होतो. लॉजिकल डाटा सामान्यतः डाटाबेसच्या संरचनेच्या अंतर्गत समाविष्ट केलेली माहिती जसे की व्यू, प्रोसिजर, फंक्शन, आणि टेबल्स असते. त्यामुळे, ह्या डाटा लॉजिकल डाटा म्हणतात, कारण हे डाटा डाटाबेसच्या अंतर्गत संरचित केलेले असतात.

यात व्हिज्यु, प्रोसिजर, टेबल आणि टेबल घेतले जाते. वापरकर्त्यांना डाटाबेसची नक्कीच उघड किंवा दुसऱ्या ठिकाणी डाटाबेसची प्रतिमा पुनर्स्थापित करण्याची इच्छा असल्यास या लॉजिकल बॅकअपस उपयुक्त आहेत. त्याच्यामुळे लॉजिकल बॅकअपपेक्षा ते डाटा गमाविण्यावरील सुरक्षिततेत थोडं कमी सुरक्षित आहे. हे केवळ संरचनात्मक माहिती प्रदान करतात. प्रत्येक आठवड्याला, पूर्ण तात्त्विक बॅकअपस करण्याची गरज आहे. लॉजिकल बॅकअपस फिजिकल बॅकअपसोबत उपयोग करण्यासाठी वापरले जातात.

लॉजिकल बॅकअपचे वैशिष्ट्ये:

- **पोर्टेबल(दुसऱ्या ठिकाणी नेण्याची क्षमता असणारे):** लॉजिकल बॅकअपस पोर्टेबल असतात आणि वेगवेगळ्या डाटाबेस सिस्टममध्ये आयात (import) करता येतात. त्यामुळे डाटा एका प्लॅटफॉर्मवरून दुसऱ्यावर हस्तांतरण (ट्रान्सफर) करणे सोपे होते.
- **डाटा फाइन-ग्रेनड(Fine grained):** विशिष्ट टेबल्स, स्कीमाज किंवा रेकॉर्ड्सची बॅकअप घेता येतो. हे बॅकअपस अधिक लवचिक असतात कारण आपण संपूर्ण डाटाबेसची बॅकअप घेण्याऐवजी फक्त आवश्यक भागांची बॅकअप घेऊ शकतो.
- **डाटा ट्रान्सफर:** लॉजिकल बॅकअपस डाटाबेसमधील डाटा इतर सिस्टममध्ये हस्तांतरण (ट्रान्सफर) करण्यासाठी वापरले जाऊ शकतात. उदा., एका डाटाबेसमधील डाटा दुसऱ्या डाटाबेसमध्ये आयात करणे.

लॉजिकल बॅकअप कसे घेतले जातात:

- **डाटा डंप:** "डाटा डंप" हे म्हणजे एका प्रणाली किंवा डाटाबेसमधून विशाल मात्रेतील डाटा निघालेलं आणि ते फाईलमध्ये किंवा इतर संग्रहात संग्रहीत केलं जातं. डाटाबेसच्या सर्व टेबल्सचा डाटा एक्सपोर्ट करून SQL स्क्रिप्ट तयार करणे.
- **स्कीमा(डाटाबेसची रचना) डंप:** फक्त डाटाबेसच्या स्कीमाची (डाटाबेसची रचना) बॅकअप घेणे.

लॉजिकल बॅकअपचे फायदे

- कधीही वापरकर्त्याला अंतिम वेळेपर्यंत पूर्ण डाटाबेस पुनर्स्थापित करण्याची आवश्यकता असताना हे उपयुक्त आहे.
- हे अधिक गुंतागुंतीचे आहे.
- फिजिकल बॅकअपचे तोटे:
- विशेष घटकांची पुनर्स्थापना साठी महत्वाचे आहे.

- फिजिकल बॅकअपशी तुलना केल्यास कमी सुरक्षित.हे केवळ संरचनात्मक माहिती पुरवते

5.4. पुनर्प्राप्ती (Data Recovery)

डाटाबेस व्यवस्थापन प्रणालीमध्ये(DBMS), डाटा रिकव्हरी म्हणजे अयशस्वी किंवा त्रुटी आल्यानंतर डाटाला सातत्यपूर्ण आणि वापरण्यायोग्य स्थितीत पुनर्संचयित करण्याच्या प्रक्रियेचा संदर्भ देते. या प्रक्रियेमध्ये कोणताही दूषित किंवा गमावलेला डाटा ओळखणे आणि दुरुस्त करणे समाविष्ट आहे, विशेषतः बॅकअप कॉपी किंवा व्यवहार लॉग वापरून डाटाबेस पुन्हा तयार करण्यासाठी अपयशी होण्यापूर्वी. डाटा रिकव्हरी हे सुनिश्चित करते की डाटाबेस विश्वसनीय राहते आणि डाटा अखंडता राखते, सिस्टम अयशस्वी झाल्यास डाटाचे नुकसान कमी करते.

रोल फॉरवर्ड हे रिकव्हरी तंत्र आहे ज्याचा वापर डाटाबेसला अद्ययावत आणण्यासाठी शेवटच्या सातत्यपूर्ण स्थितीपासून झालेले सर्व व्यवहार लागू करून वापरला जातो. कल्पना करा की तुम्ही एक पुस्तक वाचत आहात आणि तुम्ही एका विशिष्ट अध्यायात थांबता. दुसऱ्या दिवशी, तुम्ही जिथे सोडले होते तिथून पुढे चालू ठेवायचे आहे. रोल फॉरवर्ड करणे म्हणजे शेवटच्या वाचलेल्या पृष्ठापासून सुरुवात करणे आणि वर्तमान पृष्ठापर्यंत पोहोचण्यासाठी आपण गमावलेल्या सर्व पृष्ठांवर जाण्यासारखे आहे. डाटाबेस व्यवस्थापन प्रणालीमध्ये(DBMS), याचा अर्थ शेवटच्या बॅकअप किंवा सेव्ह पॉइंटपासून व्यवहार लॉगमध्ये रेकॉर्ड केलेले सर्व व्यवहार लागू करणे. हे सुनिश्चित करते की डाटाबेस त्यात केलेले सर्व बदल प्रतिबिंबित करतो, त्यास नवीनतम स्थितीत आणतो.

रोलबॅक रिकव्हरी तंत्र, रेखांकनातील चूक सुधारण्यासाठी इरेजर वापरण्यासारखे आहे. जर तुमच्या लक्षात आले की तुम्ही तुमच्या रेखांकनामध्ये चूक केली आहे, तर तुम्ही चूक पूर्ववत करण्यासाठी आणि मागील स्थितीवर परत जाण्यासाठी इरेजर वापरता. त्याचप्रमाणे, डाटाबेस व्यवस्थापन प्रणालीमध्ये(DBMS) मध्ये, व्यवहारात त्रुटी आढळल्यास किंवा अपूर्ण असल्यास, त्या व्यवहाराद्वारे केलेले सर्व बदल रोलबॅक रिव्हर्स किंवा "रोलबॅक" करतात. हे कोणतेही चुकीचे किंवा अपूर्ण बदल पूर्ववत करून डाटाबेसची सातत्य आणि अचूकता राखण्यास मदत करते.

5.5. प्रगत डाटाबेस संकल्पनांचा आढावा (Overview of Advanced database concepts):

5.5.1 डाटा वेअरहाऊस (Data Warehouse)

डाटा वेअरहाऊस(डाटाचे कोठार किंवा भांडार) हे एक किंवा अधिक विषम स्त्रोतांकडून एकत्रित डाटाचे केंद्रीय भांडार आहे. हे व्यवहार प्रक्रियेसाठी ऐवजी केरी आणि विश्लेषणासाठी डिझाइन केले आहे. डाटा वेअरहाऊसचे घटक आणि कार्ये यांचे तपशीलवार येथे आहे:

- **एकत्रीकरण थर (इंटिग्रेशन लेयर) :** डाटा वेअरहाऊस व्यवहार ट्रान्झॅक्शनल डाटाबेस, CRM सिस्टीम, ERP सिस्टीम, स्प्रेडशीट्स इ. यांसारख्या अनेक स्त्रोतांकडून डाटा एकत्रित करतात. या एकत्रीकरण प्रक्रियेमध्ये सातत्य आणि अचूकता सुनिश्चित करण्यासाठी डाटा साफ करणे, परिवर्तन करणे आणि प्रमाणित करणे समाविष्ट आहे.
- **स्टोरेज(डाटा सुरक्षित ठेवण्याची जागा):** डाटा वेअरहाऊस विश्लेषणात्मक प्रश्नांसाठी प्रभावी केलेल्या संरचित स्वरूपामध्ये मोठ्या प्रमाणात डाटा संग्रहित करतात. सामान्यतः, यामध्ये Oracle, SQL Server किंवा PostgreSQL सारखी रिलेशनल डाटाबेस मॅनेजमेंट सिस्टम (RDBMS) समाविष्ट असते. तथापि, काही डाटा वेअरहाऊस चांगल्या कामगिरीसाठी स्तंभीय डाटाबेस सारख्या पर्यायी स्टोरेज तंत्रज्ञानाचा वापर करतात.
- **डाटा मॉडेलिंग:** वेअरहाऊसमधील डाटा डायमेंशनल मॉडेलिंग किंवा स्टार स्कीमा वापरून आयोजित केला जातो. या मॉडेलमध्ये परिमाण सारण्यांनी वेढलेल्या तथ्य सारण्यांचा समावेश आहे. तथ्य सारण्यांमध्ये व्यवसाय मेट्रिक्स किंवा मोजमाप असतात, तर आयाम सारण्यांमध्ये डाटाबद्दल वर्णनात्मक गुणधर्म असतात. ही रचना कार्यक्षम केरी आणि विश्लेषण सुलभ करते.

- **ऐतिहासिक डाटा:** डाटा वेअरहाउसेस वाचविलेले डाटा विस्तृत कालावधीच्या लांबीवर साठवतात, ज्यामध्ये ट्रेंड विश्लेषण, भविष्यवाणी आणि वेळोवेळीची प्रदर्शन असतात. हा ऐतिहासिक दृष्टिकोन निर्णयात्मक क्रियाकलापांसाठी आणि धोरणात्मक (स्ट्रेटेजिक) नियोजनासाठी महत्वाचा आहे.
- **प्रश्न आणि विश्लेषण:** डाटा वेअरहाउसच्या मुख्य कामांमध्ये एक म्हणजे प्रश्नांचे आणि विश्लेषणाचे अनुमतीत करणे. वापरकर्त्यांना विस्तृत SQL प्रश्न प्रविष्ट करण्याची किंवा व्यावसायिक बुद्धिमत्तेचा (BI) उपकरणे वापरून अहवाल, डॅशबोर्ड, आणि विज्ञानांची उत्पादने निर्मिती करणे शक्य आहे. या उपकरणांमध्ये अदान-प्रदान विचारणा, ड्रिल-डाऊन, आणि डाटाची गाळणी आणि काटणी करण्याची समर्थन केली जाते.
- **डाटा रूपांतरण आणि लोडिंग (ईटीएल):** डाटा वेअरहाउसेमध्ये डाटा स्थानांतरण, रूपांतरण, आणि लोडिंग (ईटीएल) प्रक्रिया अगोदर सापडते. निघाली, रूपांतरण स्थानांतरणाचा प्रक्रिया आणि डाटा बेजता आणि अदाणीत करणे आणि तो वेअरहाउसमध्ये स्थानांतरित करणे येते. ईटीएल टूल्स ही प्रक्रिया स्वचालित करतात, ज्यामध्ये डाटा गुणवत्ता आणि संघटन पात्रता सुनिश्चित होतात.
- **मेटाडाटा व्यवस्थापन:** डाटा वेअरहाउसेस मेटाडाटा ठेवतात, ज्यात डाटा संरचना, आणि आता अंतर्निहित डाटा संदर्भ असते. मेटाडाटा डाटा स्रोतांच्या माहितीच्या, डाटा मानकीकरणाच्या, डाटा परिपाटीच्या आणि डाटा वापराच्या विषयांच्या माहिती घेते. त्यामुळे वापरकर्त्यांना डाटाची अर्थ आणि संदर्भाची समज मदत होते, प्रश्नांच्या अपशिष्टीकरण आणि डाटा व्यवस्थापनाच्या मार्गदर्शनात मदत करते.

5.5.2 डाटा लेक्स (Data Lakes)

डाटा लेक हे एक केंद्रीकृत भंडारण असते, ज्यामध्ये आपल्याला किंवा आपल्या सर्व संरचित आणि असंरचित डाटा सर्व स्तरावर ठेवून घेता येते. पारंपारिक डाटा वेअरहाउससारख्या अंशांसाठी डाटा प्रक्रिया आणि संरचनित करणे आवश्यक असल्याचं, डाटा लेक सामर्थ्यवान असतो आणि स्तरानुसार असंरचित डाटा ठेवण्यास परवानगी देतो. येथे एक डाटा लेकच्या घटकांचं आणि कामगिरीचं विस्तृत स्वरूपाचं वर्णन आहे:

- **भंडारण(storage):** डाटा लेक विकसित केलेलं असतं, जसे हडुप वितरित फाईल प्रणाली (HDFS), अमेझॉन एस 3 व्यवस्थावरील ओब्जेक्ट भंडारण किंवा अज्यूर डाटा लेक भंडारण(Azure Data Lake Storage). या प्लेटफॉर्मस सापडलेलं माहितीस विविध स्वरूपात, संरचित, अर्ध-संरचित आणि असंरचित डाटा संग्रहित करू शकतात.
- **अपरिपक्व(रॉ) डाटा संचयन:** डाटा लेकमध्ये, डाटा विशेष बिन्दू विना त्याच्या मूळ फॉर्मत संग्रहित केलं जातं. हा संरचन पारंपारिक डाटाबेस किंवा डाटा वेअरहाउससारख्या डेटाच्या संचयनाच्या वेळी किंवा डाटा लेकच्या डाटावरील स्थानिकीकरणात विनाबांधित असतं. रॉ डाटा संचयन निःसंदेह माहितीच्या विशाल मागणी आणि पुनःप्रोसेसिंग किंवा विशेषतः डाटा विश्लेषण कार्यासाठी प्राधान्य देतं.
- **डाटा कॅटलॉग(डाटा यादी):** डाटा लेकमध्ये अनेकदा डाटा कॅटलॉग किंवा मेटाडाटा भांडाराचा समावेश असतो जो उपलब्ध डाटा मालमत्तेची शोधण्यायोग्य यादी प्रदान करतो. हे कॅटलॉग वापरकर्त्यांना तलावामध्ये संग्रहित डाटा शोधण्यात, समजून घेण्यास आणि ऍक्सेस करण्यात मदत करते. यात डाटा वंश, डाटा गुणवत्ता, डाटा मालकी आणि वापर आकडेवारी, डाटा प्रशासन आणि डाटा व्यवस्थापन क्षमता वाढवणे यासारख्या मेटाडाटा समाविष्ट असू शकतात.
- **डाटा प्रोसेसिंग आणि अॅनालिटिक्स:** डाटा लेक बॅच प्रोसेसिंग, रिअल-टाइम स्ट्रीम प्रोसेसिंग, मशीन लर्निंग आणि प्रगत विश्लेषणासह डाटा प्रोसेसिंग आणि अॅनालिटिक्स वर्कलोड्सच्या विस्तृत श्रेणीला समर्थन देतात. वापरकर्ते सरोवरात साठवलेल्या डाटाचे विश्लेषण आणि अंतर्दृष्टी मिळवण्यासाठी

Apache Spark, Apache Flink, Hadoop MapReduce आणि TensorFlow सारख्या विविध प्रोसेसिंग फ्रेमवर्क आणि टूल्सचा लाभ घेऊ शकतात.

- **डाटा गव्हर्नन्स आणि सुरक्षा:** डाटा लेकमधील डाटाचे व्यवस्थापन आणि संरक्षण करण्यासाठी प्रभावी डाटा प्रशासन आणि सुरक्षा यंत्रणा आवश्यक आहेत. यामध्ये डाटा गोपनीयता, गोपनीयता आणि नियामक अनुपालन सुनिश्चित करण्यासाठी प्रवेश नियंत्रणे, एन्क्रिप्शन, प्रमाणीकरण, अधिकृतता, ऑडिटिंग आणि अनुपालन उपाय समाविष्ट आहेत.
- **विश्लेषण साधनांचे एकीकरण(Integration with Analytical Tools):** डाटा लेक विविध विश्लेषणात्मक साधने आणि प्लॅटफॉर्मसह एकत्रित होतात, ज्यामध्ये डाटा व्हिज्युअलायझेशन टूल्स, बिझनेस इंटेलिजेंस (BI) प्लॅटफॉर्म, डाटा सायन्स टूल्स आणि डाटा इंटीग्रेशन टूल्स यांचा समावेश होतो. ही इंटरऑपरेबिलिटी वापरकर्त्यांना अत्याधुनिक विश्लेषण करण्यास, अहवाल तयार करण्यास आणि डाटा लेकच्या सामग्रीवर आधारित व्हिज्युअलायझेशन तयार करण्यास सक्षम करते.
- **खर्च-प्रभावीता:** डाटा तलाव पारंपारिक डाटा वेअरहाऊसिंग पद्धतींच्या तुलनेत मोठ्या प्रमाणात डाटा संचयित आणि विश्लेषित करण्यासाठी एक किफायतशीर उपाय देतात. ते स्केलेबल, कमी किमतीच्या स्टोरेज सोल्यूशन्स आणि ओपन-सोर्स तंत्रज्ञानाचा फायदा घेतात, ज्यामुळे संस्थांना लवचिकता आणि स्केलेबिलिटी राखून आर्थिकदृष्ट्या डाटा संग्रहित आणि प्रक्रिया करण्याची परवानगी मिळते.

5.5.3. डाटा मायनिंग (Data Mining)

डाटा मायनिंग ही उपयुक्त माहिती आणि अंतर्दृष्टी काढण्यासाठी मोठ्या डाटासेटमधील नमुने, सहसंबंध, विसंगती आणि ट्रेंड शोधण्याची प्रक्रिया आहे. यात डाटाचे विश्लेषण आणि अर्थ लावण्यासाठी विविध सांख्यिकीय, गणितीय आणि मशीन लर्निंग तंत्रांचा वापर करणे समाविष्ट आहे. डाटा मायनिंगमध्ये सामील असलेल्या घटक आणि प्रक्रियांचे तपशीलवार स्पष्टीकरण येथे आहे:

- **डाटा संकलन(Data Collection):** डाटा मायनिंगची पहिली पायरी म्हणजे डाटाबेस, डाटा वेअरहाऊस, इंटरनेट, सेन्सर्स किंवा व्यवहार रेकॉर्ड यासारख्या विविध स्रोतांकडून संबंधित डाटा गोळा करणे. हा डाटा संरचित, अर्ध-संरचित किंवा असंरचित असू शकतो.
- **डाटा क्लीनिंग आणि प्रीप्रोसेसिंग[परिपक्व(स्वच्छ)] (Data Cleaning):** गोळा केलेल्या कच्च्या डाटामध्ये त्रुटी, गहाळ मूल्ये किंवा विसंगती असू शकतात. डाटा क्लीनिंगमध्ये डाटा गुणवत्ता सुनिश्चित करण्यासाठी या समस्या ओळखणे आणि दुरुस्त करणे समाविष्ट आहे. प्रीप्रोसेसिंगमध्ये डाटा सामान्यीकरण, परिवर्तन आणि विश्लेषणासाठी डाटा तयार करण्यासाठी वैशिष्ट्य निवड यासारख्या कार्यांचा समावेश होतो.
- **एक्सप्लोरेटरी डाटा अॅनालिसिस (अन्वेषणात्मक विश्लेषण) (EDA):** EDA मध्ये सांख्यिकीय आणि व्हिज्युअलायझेशन तंत्रांचा वापर करून डाटासेटची वैशिष्ट्ये शोधणे आणि सारांशित करणे समाविष्ट आहे. डाटा मायनिंग अल्गोरिदम लागू करण्यापूर्वी डाटामधील वितरण, संबंध आणि नमुने समजून घेण्यास हे मदत करते.
- **वैशिष्ट्य निवड आणि अभियांत्रिकी:** वैशिष्ट्य निवडीमध्ये मॉडेलच्या अंदाज शक्तीमध्ये योगदान देणारी सर्वात संबंधित वैशिष्ट्ये किंवा चल निवडणे समाविष्ट असते. वैशिष्ट्य अभियांत्रिकीमध्ये मॉडेल कार्यप्रदर्शन सुधारण्यासाठी नवीन वैशिष्ट्ये तयार करणे किंवा विद्यमान वैशिष्ट्ये बदलणे समाविष्ट आहे.
- **मॉडेल निवड आणि प्रशिक्षण:** डाटा मायनिंगमध्ये विविध अल्गोरिदम लागू करणे समाविष्ट आहे, जसे की निर्णय झाडे, न्यूरल नेटवर्क्स, सपोर्ट वेक्टर मशीन्स, क्लस्टरिंग अल्गोरिदम किंवा असोसिएशन नियम मायनिंग, डाटाचे विश्लेषण करण्यासाठी आणि नमुना काढण्यासाठी. मॉडेलची निवड समस्येचे स्वरूप

आणि डाटासेटच्या वैशिष्ट्यांवर आधारित आहे. नमुने आणि नातेसंबंध जाणून घेण्यासाठी ऐतिहासिक डाटा वापरून मॉडेलला प्रशिक्षण दिले जाते.

- **मूल्यमापन आणि प्रमाणीकरण:** एकदा मॉडेल प्रशिक्षित झाल्यानंतर, त्यांच्या कामगिरीचे मूल्यांकन करण्यासाठी त्यांचे मूल्यमापन आणि प्रमाणीकरण करणे आवश्यक आहे. यामध्ये डाटासेटला प्रशिक्षण आणि चाचणी संचांमध्ये विभाजित करणे, क्रॉस-व्हॅलिडेशन किंवा मॉडेलचे कार्यप्रदर्शन मोजण्यासाठी अचूकता, अचूकता, रिकॉल किंवा F1-स्कोअर यासारख्या मेट्रिक्सचा वापर करणे समाविष्ट आहे.
- **उपयोजन आणि देखरेख:** डाटा मायनिंग प्रक्रिया पूर्ण झाल्यानंतर आणि मॉडेलचे प्रमाणीकरण झाल्यानंतर, ते भविष्य सांगण्यासाठी किंवा निर्णय घेण्यास चालना देण्यासाठी उत्पादन वातावरणात तैनात केले जाऊ शकतात. कालांतराने उपयोजित मॉडेलच्या कार्यप्रदर्शनाचे निरीक्षण करणे आणि डाटा किंवा व्यवसाय वातावरणातील बदलांशी जुळवून घेण्यासाठी त्यांना वेळोवेळी पुन्हा प्रशिक्षित करणे आवश्यक आहे.

5.5.4. बिग डाटा (Big Data)

बिग डाटा संरचित, अर्ध-संरचित आणि असंरचित डाटाच्या मोठ्या प्रमाणाचा संदर्भ देते जे दररोजच्या आधारावर संस्थांना भरते. हा डाटा सोशल मीडिया, सेन्सर्स, मोबाइल डिव्हाइसेस, इंटरनेट व्यवहार आणि बरेच काही यासह विविध स्रोतांकडून येतो. "बिग डाटा" हा शब्द केवळ डाटाच्या आकाराविषयी नाही तर डाटाचा वेग, विविधता आणि जटिलता देखील समाविष्ट करतो.

बिग डाटाच्या मुख्य घटकांचे आणि वैशिष्ट्यांचे येथे तपशीलवार स्पष्टीकरण आहे:

- **क्षमता(Volume):** बिग डाटा त्याच्या संपूर्ण व्हॉल्यूमद्वारे दर्शविला जातो. पारंपारिक डाटा प्रोसेसिंग साधने आणि तंत्रे दर सेकंदाला प्रचंड प्रमाणात तयार होणारा डाटा हाताळण्यासाठी अपुरी आहेत. संस्था टेराबाइट्स, पेटाबाइट्स किंवा अगदी एक्साबाइट्स डाटाचा व्यवहार करतात आणि अशा मोठ्या व्हॉल्यूमचे व्यवस्थापन आणि विश्लेषण करणे हे एक महत्त्वाचे आव्हान आहे.
- **वेग(Velocity):** बिग डाटाचे आणखी एक परिभाषित वैशिष्ट्य म्हणजे त्याचा वेग, ज्या गतीने डाटा व्युत्पन्न केला जातो, गोळा केला जातो आणि त्यावर प्रक्रिया केली जाते. इंटरनेट, सोशल मीडिया आणि IoT उपकरणांच्या प्रसारासह, डाटा अभूतपूर्व दराने व्युत्पन्न केला जातो. वेळेवर अंतर्दृष्टी मिळविण्यासाठी आणि माहितीपूर्ण निर्णय घेण्यासाठी डाटा प्रवाहांची रिल-टाइम किंवा जवळ-रिल-टाइम प्रक्रिया आवश्यक आहे.
- **विविधता(Variety):** बिग डाटा संरचित, अर्ध-संरचित आणि असंरचित डाटासह विविध स्वरूपांमध्ये येतो. संरचित डाटा चांगल्या-परिभाषित स्कीमाचे अनुसरण करतो आणि सामान्यतः रिलेशनल डाटाबेसमध्ये संग्रहित केला जातो. JSON किंवा XML सारख्या अर्ध-संरचित डाटामध्ये काही संस्थात्मक गुणधर्म आहेत परंतु कठोर स्कीमा नाही. मजकूर दस्तऐवज, प्रतिमा, व्हिडिओ आणि सोशल मीडिया पोस्ट यांसारख्या असंरचित डाटाची पूर्वनिर्धारित रचना नसते.
- **सत्यता(Veracity):** सत्यता म्हणजे डाटाची विश्वासार्हता किंवा विश्वासार्हता. मोठ्या डाटामध्ये बऱ्याचदा अचूकता आणि सुसंगततेच्या विविध स्तरांसह विविध स्रोतांकडील डाटा समाविष्ट असतो. डाटा गुणवत्तेशी संबंधित समस्या, जसे की गहाळ मूल्ये, आउटलियर किंवा त्रुटी, डाटामधून मिळवलेल्या विश्लेषणाच्या आणि अंतर्दृष्टीच्या वैधतेवर परिणाम करू शकतात. मोठ्या डाटाची सत्यता सुनिश्चित करण्यासाठी डाटा साफ करणे, प्रमाणीकरण आणि गुणवत्ता हमी तंत्रे आवश्यक आहेत.
- **मूल्य(Value):** मोठ्या डाटाचे अंतिम उद्दिष्ट संकलित केलेल्या डाटाच्या मोठ्या प्रमाणावर मूल्य आणि कृती करण्यायोग्य अंतर्दृष्टी प्राप्त करणे आहे. मोठ्या डाटाचे विश्लेषण करून, संस्था नमुने, ट्रेंड, सहसंबंध आणि लपलेले अंतर्दृष्टी उघड करू शकतात जे व्यवसाय निर्णय घेऊ शकतात, ऑपरेशन्स

ऑप्टिमाइझ करू शकतात, ग्राहक अनुभव सुधारू शकतात आणि नवीन संधी ओळखू शकतात. मोठ्या डाटामधून मूल्य काढण्यासाठी प्रगत विश्लेषणे, मशीन लर्निंग आणि डाटा व्हिज्युअलायझेशन तंत्र आवश्यक आहे.

बिग डाटाचे प्रभावीपणे व्यवस्थापन आणि विश्लेषण करण्यासाठी, संस्था यासह तंत्रज्ञान, प्लॅटफॉर्म आणि फ्रेमवर्कच्या संयोजनावर अवलंबून असतात:

- **वितरित संगणन(Distributed Computing):** Apache Hadoop आणि Apache Spark सारखे वितरित संगणन फ्रेमवर्क कमोडिटी हार्डवेअरच्या क्लस्टरमध्ये मोठ्या डाटासेटची समांतर प्रक्रिया सक्षम करतात, ज्यामुळे स्केलेबल आणि किफायतशीर डाटा प्रोसेसिंग करता येते.
- **NoSQL डाटाबेस (NoSQL Database):** NoSQL डाटाबेस, जसे की MongoDB, Cassandra आणि Couchbase, उच्च स्केलेबिलिटी आणि लवचिकतेसह मोठ्या प्रमाणात असंरचित आणि अर्ध-संरचित डाटा हाताळण्यासाठी डिझाइन केलेले आहेत.
- **स्ट्रीमिंग ॲनालिटिक्स(Streaming Analytics):** Apache Kafka आणि Apache Flink सारखे स्ट्रीमिंग विश्लेषण प्लॅटफॉर्म रीअल-टाइम प्रोसेसिंग आणि डाटा स्ट्रीमचे विश्लेषण सक्षम करतात, ज्यामुळे संस्थांना अंतर्दृष्टी काढता येते आणि येणाऱ्या डाटावर त्वरित कारवाई करता येते.
- **डाटा गव्हर्नन्स आणि सुरक्षा(Data Governance and Security):** प्रभावी डाटा प्रशासन आणि सुरक्षा पद्धती मोठ्या डाटा वातावरणात संवेदनशील डाटाचे व्यवस्थापन आणि संरक्षण करण्यासाठी आवश्यक आहेत. यामध्ये डाटा गोपनीयता, सुरक्षा आणि नियामक अनुपालन सुनिश्चित करण्यासाठी प्रवेश नियंत्रणे, एन्क्रिप्शन, प्रमाणीकरण, ऑडिटिंग आणि अनुपालन उपाय समाविष्ट आहेत.

5.5.5. मोंगो (Mongo DB)

मोंगोडीबी(Mongo DB) हा एक लोकप्रिय, मुक्त-स्रोत(open source) NoSQL डाटाबेस आहे जो विविध प्रकारचे डाटा हाताळण्यासाठी उच्च कार्यक्षमता, स्केलेबिलिटी(वाढत्या वर्कलोडमध्ये चांगली कामगिरी करण्याची क्षमता) आणि लवचिकता प्रदान करतो. पारंपारिक रिलेशनल डाटाबेसेसच्या विपरीत, मोंगोडीबी एक दस्तऐवज-देणार डाटा मॉडेल फॉलो करते, निश्चित स्कीमासह कठोर टेबलांऐवजी लवचिक, JSON-सारख्या दस्तऐवजांमध्ये डाटा संग्रहित करते.

मोंगोडीबीची प्रमुख वैशिष्ट्ये आणि घटकांचे तपशीलवार स्पष्टीकरण येथे आहे:

- **दस्तऐवज-ओरिएंटेड स्टोरेज(Document-Oriented Storage):** मोंगोडीबी(Mongo DB) लवचिक, JSON सारख्या दस्तऐवजांमध्ये डाटा संग्रहित करते ज्याला BSON (बायनरी JSON) म्हणतात. प्रत्येक दस्तऐवजाची स्वतःची अनन्य रचना असू शकते, ज्यामुळे विकसक जटिल श्रेणीबद्ध संबंध आणि नेस्टेड डाटा स्ट्रक्चर्सचे सहज प्रतिनिधित्व करू शकतात. हा स्कीमा-लेस दृष्टिकोन डाटा मॉडेलिंग आणि स्कीमा डिझाइनमध्ये लवचिकता आणि चपळता प्रदान करतो.
- **संग्रह आणि दस्तऐवज(Collections and Documents):** मोंगोडीबी (Mongo DB)मधील डाटा संग्रहांमध्ये आयोजित केला जातो, जो रिलेशनल डाटाबेसमधील सारण्यांप्रमाणे(table) असतो. प्रत्येक संग्रहामध्ये एकाधिक दस्तऐवज असतात, जे वैयक्तिक रेकॉर्ड किंवा डाटा एंट्री असतात. दस्तऐवज संरचनेत भिन्न असू शकतात आणि त्यात अरे, नेस्टेड दस्तऐवज आणि इतर जटिल डाटा प्रकार असू शकतात.
- **स्कीमा लवचिकता(Flexible Schema):** MongoDB चे स्कीमा-लेस डिझाइन डायनॅमिक आणि विकसित स्कीमांना अनुमती देते. विकसक संग्रहातील इतर दस्तऐवजांना प्रभावित न करता दस्तऐवजांमधून फील्ड जोडू किंवा काढू शकतात. ही लवचिकता विशेषतः अशा परिस्थितीत उपयुक्त

आहे जिथे डाटा स्कीमा वेगाने विकसित होतात किंवा जिथे डाटा वेगवेगळ्या रेकॉर्डमध्ये लक्षणीयरीत्या बदलतो.

- **उच्च कार्यप्रदर्शन आणि स्केलेबिलिटी(High Performance and Scalability):** मोंगोडीबी (Mongo DB) उच्च कार्यप्रदर्शन आणि स्केलेबिलिटीसाठी डिझाइन केलेले आहे, मोठ्या प्रमाणात डाटा आणि उच्च थ्रूपुट वर्कलोड हाताळण्यास सक्षम आहे. हे मेमरी-मॅप केलेले स्टोरेज आणि असिंक्रोनस(दोन किंवा अधिक घटना/ऑब्जेक्ट्समधील संबंध जे एकाच प्रणालीमध्ये परस्परसंवाद करतात परंतु पूर्वनिर्धारित अंतराने होत नाहीत) I/O ऑपरेशन्स कमी-विलंब वाचन आणि लेखन ऑपरेशन्स साध्य करण्यासाठी वापरते. MongoDB चे वितरित आर्किटेक्चर एकाधिक नोड्समध्ये क्षेत्रीय स्केलिंगसाठी परवानगी देते, स्टोरेज क्षमता आणि प्रक्रिया शक्तीचा अखंड विस्तार सक्षम करते.
- **रिच क्वेरी लॅंग्वेज(Rich Query Language):** मोंगोडीबी (Mongo DB) एका शक्तिशाली क्वेरी भाषेचे समर्थन करते जी विकासकांना फिल्टरिंग, क्रमवारी, एक्झिक्युशन आणि भौगोलिक प्रश्नांसह जटिल क्वेरी करण्यास अनुमती देते. JSON सारखी वाक्यरचना वापरून क्वेरी व्यक्त केल्या जाऊ शकतात आणि क्वेरी कार्यप्रदर्शन सुधारण्यासाठी निर्देशांकांचा फायदा घेऊ शकतात.
- **अनुक्रमणिका(Indexing):** मोंगोडीबी (Mongo DB) एकल-फील्ड, कंपाउंड, मल्टी-की आणि भू-स्थानिक निर्देशांकांसह विविध प्रकारच्या निर्देशांकांना समर्थन देते. निर्देशांक कार्यक्षम डाटा पुनर्प्राप्ती आणि फिल्टरिंग सुलभ करून क्वेरी कार्यप्रदर्शन सुधारण्यात मदत करतात. डेव्हलपर वारंवार ऍक्सेस केलेल्या डाटासाठी क्वेरी एक्झिक्यूशन ऑप्टिमाइझ करण्यासाठी विशिष्ट फील्डवर अनुक्रमणिका तयार करू शकतात.
- **प्रतिकृती आणि उच्च उपलब्धता(Replication and High Availability):** मोंगोडीबी (Mongo DB) प्रतिकृतीसाठी अंगभूत समर्थन प्रदान करते, ज्यामुळे दोष सहिष्णुता आणि उच्च उपलब्धतेसाठी क्लस्टरमध्ये एकाधिक नोड्समध्ये डाटाची प्रतिकृती तयार केली जाऊ शकते. नोड निकामी झाल्यास प्रतिकृती डाटा टिकाऊपणा आणि स्वयंचलित फेलओव्हर सुनिश्चित करते, डाटाबेस सिस्टमचे सतत ऑपरेशन सुनिश्चित करते.
- **सुरक्षा वैशिष्ट्ये(Security Features):** अनाधिकृत प्रवेशापासून डाटाचे संरक्षण करण्यासाठी आणि डाटा गोपनीयता आणि अखंडता सुनिश्चित करण्यासाठी MongoDB मजबूत सुरक्षा वैशिष्ट्ये ऑफर करते. या वैशिष्ट्यांमध्ये प्रमाणीकरण, भूमिका-आधारित प्रवेश नियंत्रण (RBAC), विश्रांती आणि संक्रमणामध्ये एन्क्रिप्शन, ऑडिटिंग आणि सूक्ष्म प्रवेश नियंत्रणे समाविष्ट आहेत.

5.5.6. डायनॅमो डीबी (DynamoDB)

DynamoDB ही Amazon Web Services (AWS) द्वारे प्रदान केलेली पूर्णपणे व्यवस्थापित NoSQL डाटाबेस सेवा आहे. हे उच्च कार्यप्रदर्शन, स्केलेबिलिटी आणि डाटामध्ये कमी विलंब प्रवेशासाठी डिझाइन केले आहे.

DynamoDB बद्दल समजून घेण्यासाठी येथे सात महत्वाचे मुद्दे आहेत:

- **पूर्णपणे व्यवस्थापित सेवा (Fully Managed Service):** DynamoDB ही पूर्णपणे व्यवस्थापित डाटाबेस सेवा आहे, म्हणजे AWS पायाभूत सुविधांची तरतूद, स्केलिंग आणि व्यवस्थापन हाताळते, ज्यामुळे विकासकांना डाटाबेस प्रशासनाऐवजी ऍप्लिकेशन डेव्हलपमेंटवर लक्ष केंद्रित करता येते. हे ऑपरेशन्स सुलभ करते आणि डाटाबेस सर्व्हरच्या व्यवस्थापनाशी संबंधित ओव्हरहेड कमी करते.
- **स्केलेबिलिटी (Scalability):** वाढत्या वर्कलोड्स आणि डाटा व्हॉल्यूमला सामावून घेण्यासाठी डायनॅमोडीबी क्षेत्रीयरीत्या स्केल करण्यासाठी डिझाइन केले आहे. हे उच्च थ्रूपुट आणि स्टोरेज आवश्यकता हाताळण्यासाठी एकाधिक विभाजनांमध्ये स्वयंचलितपणे डाटा वितरित करते. जसजशी

मागणी वाढते तसतसे, DynamoDB कोणत्याही डाउनटाइम किंवा कार्यक्षमतेत घट न होता अखंडपणे वाढ किंवा खाली करू शकते.

- **कार्यप्रदर्शन (Performance):** DynamoDB वाचन आणि लेखन ऑपरेशन्ससाठी सिंगल-अंकी मिलिसेकंद लेटन्सी ऑफर करते, ज्यांना डाटामध्ये कमी-लेटन्सी ऍक्सेस आवश्यक आहे अशा ऍप्लिकेशन्ससाठी ते योग्य बनवते. स्टोरेजसाठी सॉलिड-स्टेट ड्राइव्हस् (SSDs) वापरून आणि विभाजन आणि कॅशिंग यंत्रणेद्वारे डाटा ऍक्सेस पॅटर्न ऑप्टिमाइझ करून उच्च कार्यक्षमता प्राप्त करते.
- **लवचिक डाटा मॉडेल (Flexible Data Model):** DynamoDB की-वॅल्यू जोड्या आणि दस्तऐवज-सदृश संरचनांवर आधारित लवचिक डाटा मॉडेलचे समर्थन करते. DynamoDB मधील प्रत्येक आयटम प्राथमिक की द्वारे अद्वितीयपणे ओळखला जातो, जी एकतर एक साधी की असू शकते किंवा विभाजन की आणि सॉर्ट की बनलेली संमिश्र की असू शकते. हे लवचिक डाटा मॉडेल विकसकांना संरचित, अर्ध-संरचित आणि असंरचित डाटा कार्यक्षमतेने संचयित आणि पुनर्प्राप्त करण्यास अनुमती देते.
- **सुसंगतता मॉडेल (Consistency Model):** DynamoDB दोन सातत्य मॉडेल ऑफर करते: मजबूत सातत्य आणि अंतिम सातत्य. मजबूत सुसंगतता हे सुनिश्चित करते की सर्व वाचन नवीनतम लेखन प्रतिबिंबित करतात, तर अंतिम सुसंगतता सर्व प्रतिकृतींमध्ये लेखनाचा अंतिम प्रसार करण्यास अनुमती देते. विकासक त्यांच्या अनुप्रयोगाच्या आवश्यकतांवर आधारित योग्य सुसंगतता मॉडेल निवडू शकतात.
- **सुरक्षा आणि अनुपालन (Security and Compliance):** DynamoDB आरामात आणि संक्रमणामध्ये डाटाचे संरक्षण करण्यासाठी मजबूत सुरक्षा वैशिष्ट्ये प्रदान करते. हे AWS की मॅनेजमेंट सर्व्हिस (KMS) वापरून विश्रांतीच्या वेळी एन्क्रिप्शन आणि ट्रान्सपोर्ट लेयर सिव्युरिटी (TLS) वापरून ट्रान्झिटमध्ये एन्क्रिप्शनचे समर्थन करते. याव्यतिरिक्त, DynamoDB AWS आयडेंटिटी आणि ऍक्सेस मॅनेजमेंट (IAM) आणि Amazon VPC एंडपॉईंट धोरणांद्वारे सूक्ष्म प्रवेश नियंत्रण ऑफर करते.
- **AWS इकोसिस्टमसह एकीकरण (Integration with AWS Ecosystem):** DynamoDB इतर AWS सेवांसह अखंडपणे समाकलित करते, जसे की AWS Lambda, Amazon API गेटवे, Amazon S3, Amazon Kinesis आणि Amazon CloudWatch. हे एकीकरण विकासकांना प्राथमिक डाटा स्टोअर म्हणून डायनामोडीबी वापरून सर्व्हरलेस ऍप्लिकेशन्स, रीअल-टाइम स्ट्रीमिंग पाइपलाइन आणि इव्हेंट-चालित आर्किटेक्चर्स तयार करण्यास अनुमती देते.

सारांश, DynamoDB ही पूर्णपणे व्यवस्थापित NoSQL डाटाबेस सेवा आहे जी स्केलेबिलिटी, उच्च कार्यप्रदर्शन, लवचिक डाटा मॉडेलिंग, मजबूत सुरक्षा आणि AWS इकोसिस्टमसह अखंड एकीकरण देते. हे वेब आणि मोबाइल ऍप्लिकेशन्स, गेमिंग, IoT, जाहिरात तंत्रज्ञान आणि रीअल-टाइम अॅनालिटिक्ससह विस्तृत वापर प्रकरणांसाठी योग्य आहे.

References:

1. <https://www.oracle.com/in/database/what-is-database/>
2. Database System Concepts by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan, 6th Edition, McGraw-Hill Education, 2010.
3. <https://www.geeksforgeeks.org/introduction-of-dbms-database-management-system-set-1/>
4. <https://www.scaler.com/topics/dbms/>
5. <https://nptel.ac.in/courses/106105175>
6. <https://www.tutorialspoint.com/sql/index.htm>
7. <https://www.w3schools.com/sql/>