|  |  |  |  |
| --- | --- | --- | --- |
| No. | Title | Points to be covered / example | Length |
| 1 | Introduction to data types. | Note: Welcome to the session on Introduction to data types which is basic but very important topic to design databases.  What we going to cover in this session/ In this session we are going to cover/  Today’s topic are / Today we are going to discuss/.  Categories of datatype numeric, string, binary  Discussion on numeric data types (int, smallint, etc) including their ranges.  Difference between dec and floot.  Discussion on String datatype (char, varchar, text, etc) non unicode  Diff between char and varchar  Discussion on nvarchar/ Unicode  Diff between varchar and nvarchar (2)  Discussion on BLOB (full form to be told) – small, bigblob, etc.  Tell them the simillraty between blob and string.  CHAR, VARCHAR, TINYINT, INT, YEAR, DATE, BOOLEAN  Char(10)  Varchar(10)  e.g. todo  e.g.  DECIMAL(7, 2) Total length of this data type is 7 of which 2 will be decimal part.  5396.17 ok value  5396.174 5396.17  5396.178 5396.18  5396.1743 5396.17  5396.1748 5396.17  5396.1788 5396.18  Note: With this we have covered all the today’s topics.  I hope you understood data types in SQL. | 15 |
| 2 | SQL Commands | Note: Welcome back to the session on SQL Commands.  What are SQL commands   1. DDL – Data Definition Language 2. DML – Data Manipulation Language 3. DCL – Data Control Language 4. TCL – Transaction Control Language 5. Constraints   DDL – Discuss – Full list of commands with brief description  DML– Discuss – Full list of commands with brief description  DCL– Discuss – Full list of commands with brief description  TCL– Discuss – Full list of commands with brief description  Discussion on types Of Constraints (in brief)   * **NOT NULL** - Ensures that a column cannot have a NULL value * **UNIQUE** - Ensures that all values in a column are different * **PRIMARY KEY** - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table * **FOREIGN KEY** - Uniquely identifies a row/record in another table. * **CHECK** - Ensures that all values in a column satisfy a specific condition. | 20 |
| 3(P1) | Database management system | Intro to this course (Ulka and saleel - before recording discuss with sir)  Pre req. – For whom this course imp (eg. Application dev, DBA’s, Data Mining  Career imp - to become DBA, Data Sciten, etc.  Why this course is easy to learn (Section of course, number of sessions, total duration course duration 30hrs.)  Examples been covered. (number of databases, tables discusse)  Note: To be record at the last, content continually updated. | 5 |
| 4(P3) | Introduction to RDBMS | TODO: | 8 |
| 5(P3) | File System vs DBMS | Concepts of Flat Files  Concepts of DBMS  Advantages and disadv of Flat Files  Advantages and disadv ofDBMS  Slides with graphics explaining the above concepts (good explainatery images)  Note: Content continually updated. | 5 |
| 6(P1) | Schema in databases | Intro to schema  Defining and creating schema  Managing and moving objects between schemas  Note: Slides required, live examples/demos ( Create Schema/database in MySQL is same, Use schema, moving objects (in MS-SQL possible) from one schema to another schema) | 10 |
| 7(P1) | Candidate and Primary key | What is a candidate key  What is a primary key  Diff between CK and PK  Points to consider to make column PK and CK | 10 |
| 8(P1) | Primary Key | Imp of PK  Examples of PK  PK data type mostly it must be INT(eg ID), (Eg email varchar PK)  PK and AUTO\_INCREMENT  NOT NULL  Using multiple columns as PK (Composite Key)  Note: With Slides and Examples (student Table R= {ID, firstName, lastName, DoB, emailID } ) | 15 |
| 9(P1) | Foreign Key | Concepts of FK  Examples of FK  Data type of PK/FK must be same  Note: With Slides and Examples (student Table R= {ID(PK), firstName, lastName, DoB, emailID } , student\_address Table, student\_hobbies table}) – insert 4-5 demo records  Structure view and Brows view  URL: <https://dev.mysql.com/doc/workbench/en/wb-creating-eer-diagram.html> |  |
| 10(P1) | Introduction to DML commands | Use of INSERT, SELECT, UPDATE, and DELETE with single table (student table)  Use of above commands with related tables (PK/FK table – student - student\_address)  Referential integrity (Directly adding a record in student\_address, deleting a student record,  Similar for student\_hobbies table)  Note: With Slides and Examples (student Table R= {ID(PK), firstName, lastName, DoB, emailID } , student\_address Table, student\_hobbies table})  Structure view and Brows view | 20 |
| 11(P2) | Working with ER models | What are ER models  Working with MySQL ER model Tool.  Benefits of documentation  Third party ER database documentations tools (toad from DELL, NaviCat)  Note: With Slides and Examples | 5-10 |
| 12(P3) | Using ER model for Project Documentations | Attributes of ER models  TODO |  |
| 13(P1) | Introduction to Relationships | Basic intro to all relationships  Note: With Slides with relationship definition and ER images | 5 |
| 14(P1) | One-to-one Relationship | Discussion of one-to-one relationships  Design intents for creating one-to-one relationships (situations like student\_address, one more example to be added)  How to create one-to-one relationship (T1-PK/T2-PK&FK)  Note: With Slides and Examples | 10 |
| 15(P1) | One-to-many Relationship | Discussion of one-to-many relationships  Design intents for creating one-to-many relationships (situations like student\_hobbies, one more example of **invoice** and **invoice\_items**) (Every one invoice must have at least one item))  How to create one-to-many relationship  Note: With Slides and Examples | 10 |
| 16(P1) | Many-to-many Relationship | Discussion of many-to-many relationships  Design intents for creating many-to-many relationships (situations like student, course, and course\_student)  How to create many-to-many relationship  Note: With Slides and Examples | 10 |
| 17(P4) | Many-to-many Relationship - 2 | Solving advance problem statements | TODO |
| 18(P2) | Introduction to Normalization | Concepts of Normalization  Reasons to normalize the database.  Brief on all normalization levels  Note: With Slides-good graphics | 10 |
| 19(P2) | Normalization | Insertion, Updation, and Deletion Anomaly  Note: With Slides and example | 5-10 |
| 20(P2) | First form of Normalization | Def of first form of normalization (1NF)  Understanding of closure of functional dependency  Understanding functional dependency and its properties.  Note: With Slides-good graphics and example | 10 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 2 | Create Table with DEFAULT property | Create Table and Default property   * CREATE TABLE temp (   ID int primary key,  firstname varchar(45),  phone int,  city varchar(10) **default** '**PUNE**'  );   * CREATE TABLE employee (   ID int primary key,  firstname varchar(45),  phone int,  city varchar(10) **default** '**PUNE**’,  salary int,  comm int,  total int **default** (salary **+** comm) // version 8.0 and above.  ); |  |
| 3 | Create Table with AUTO\_INCREMENT Property | AUTO\_INCREMENT  CREATE TABLE employee (  ID int auto\_increment primary key,  firstname varchar(40),  salary int,  commission int,  total int default (salary + commission)  ) ;   * INSERT INTO employee VALUES(NULL, 'ram', 4700, NULL, default); * INSERT INTO employee VALUES(0, 'pankaj', 3400, 400 , default); * INSERT INTO employee VALUES(100, 'rajan', 3200, NULL , default); * INSERT INTO employee VALUES(NULL, 'ninad', 2600, 0 , default); * INSERT INTO employee VALUES(0, 'omkar', 4500, 300 , default); |  |
| 4 | Create Table with ZEROFILL property | ZEROFILL property  CREATE TABLE employee (  ID int zerofill auto\_increment primary key,  firstName varchar(40),  salary int,  commission int,  total int default (salary + commission)  ) ;   * INSERT INTO employee VALUES(NULL, 'ram', 4700, NULL, default); * INSERT INTO employee VALUES(0, 'pankaj', 3400, 400 , default); * INSERT INTO employee VALUES(100, 'rajan', 3200, NULL , default); * INSERT INTO employee VALUES(NULL, 'ninad', 2600, 0 , default); * INSERT INTO employee VALUES(0, 'omkar', 4500, 300 , default); * INSERT INTO employee VALUES (-200, 'rahul', 3000, 300 , default); * SELECT \* FROM employee; * SELECT ID, length(ID), salary, length(salary) FROM employee; |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

file name**Introduction to Data-Types V1.0.mp4**