SQL
Session - 02

Learning Objectives

SQL - Datatypes

- A data type defines what kind of value a column can hold: integer data, character data, monetary data, date and time data, binary strings, and so on.
- SQL Data Types
  - Each column in a database table is required to have a name and a data type.
  - An SQL developer must decide what type of data that will be stored inside each column when creating a table. The data type is a guideline for SQL to understand what type of data is expected inside of each column, and it also identifies how SQL will interact with the stored data.

Note: Data types might have different names in different database. And even if the name is the same, the size and other details may be different! **Always check the documentation!**

Datatypes – Text

Data type	Description
CHAR(size)	Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to 255 characters
VARCHAR(size)	Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to 255 characters. Note: If you put a greater value than 255 it will be converted to a TEXT type
TINYTEXT	Holds a string with a maximum length of 255 characters
TEXT	Holds a string with a maximum length of 65,535 characters
BLOB	For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data
MEDIUMTEXT	Holds a string with a maximum length of 16,777,215 characters
MEDIUMBLOB	For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data
LONGTEXT	Holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
ENUM(x,y,z,etc.)	Let you enter a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. Note: The values are sorted in the order you enter them. You enter the possible values in this format: ENUM('X','Y','Z')
SET	Similar to ENUM except that SET may contain up to 64 list items and can store more than one choice

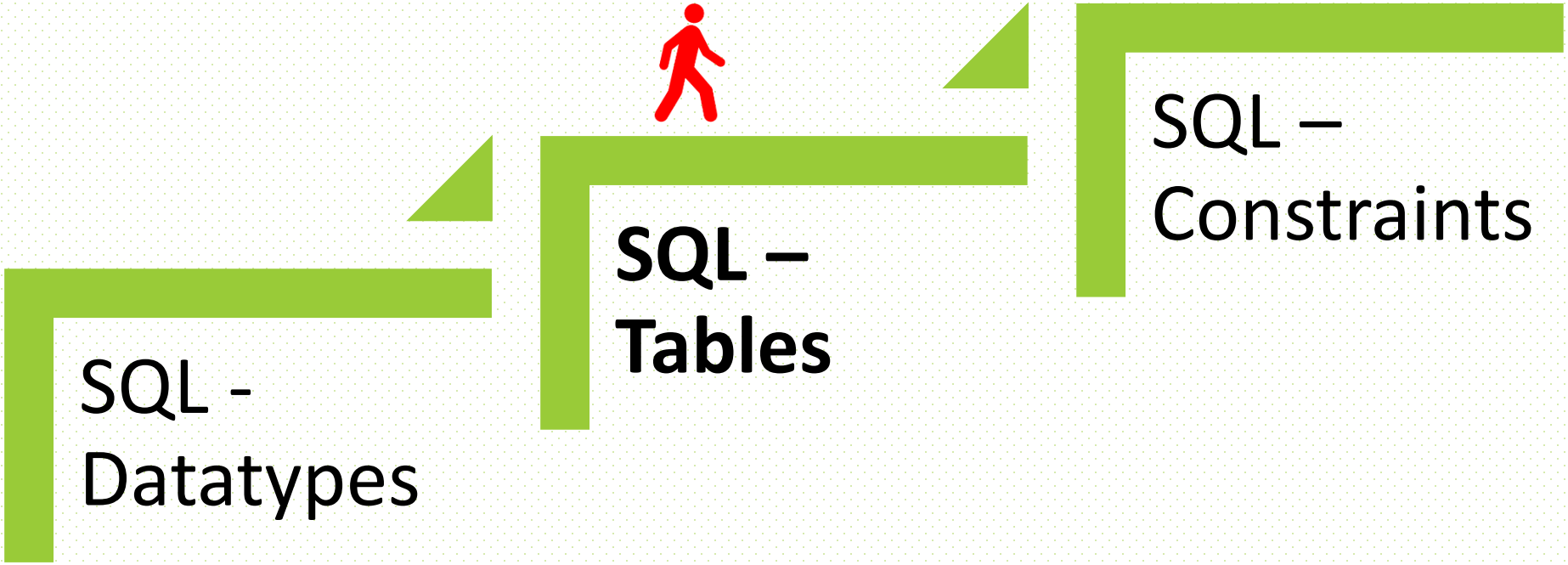
Datatypes – Number

Data type	Description
TINYINT(size)	-128 to 127 normal. 0 to 255 UNSIGNED*. The maximum number of digits may be specified in parenthesis
SMALLINT(size)	-32768 to 32767 normal. 0 to 65535 UNSIGNED*. The maximum number of digits may be specified in parenthesis
MEDIUMINT(size)	-8388608 to 8388607 normal. 0 to 16777215 UNSIGNED*. The maximum number of digits may be specified in parenthesis
INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
BIGINT(size)	-9223372036854775808 to 9223372036854775807 normal. 0 to 18446744073709551615 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE stored as a string , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

Datatypes – Date

Data type	Description
DATE()	A date. Format: YYYY-MM-DD**Note:** The supported range is from '1000-01-01' to '9999-12-31'
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP()	*A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC
TIME()	A time. Format: HH:MI:SS **Note:** The supported range is from '-838:59:59' to '838:59:59'
YEAR()	A year in two-digit or four-digit format. **Note:** Values allowed in four-digit format: 1901 to 2155. Values allowed in two-digit format: 70 to 69, representing years from 1970 to 2069

Learning Objectives



SQL - CREATE TABLE

Syntax

```
CREATE TABLE <table-name> (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

SQL - CREATE TABLE Using Another Table

Syntax

```
CREATE TABLE <table-name> AS
    SELECT column1, column2,...
    FROM <existing-table-name>
    WHERE ....;
```

SQL – View Table Description

Syntax

DESCRIBE *<table-name>;*

SQL – ALTER TABLE (Add Column)

Syntax

ALTER TABLE <*table-name*>
ADD <*column-name*> *datatype*;

SQL – ALTER TABLE (Modify Column)

Syntax

ALTER TABLE <*table-name*>
MODIFY COLUMN <*column-name*> <*datatype*>;

SQL – ALTER TABLE (Drop Column)

Syntax

ALTER TABLE <*table-name*>
DROP <*column-name*>;

SQL – DROP/TRUNCATE TABLE

Syntax

DROP TABLE <*table-name*>;

TRUNCATE TABLE <*table-name*>;

SQL - Tasks

Task 1 – Create table customers with fields – (id (int), name(varchar), mobile(int), address(varchar), city (varchar), country (varchar), pin (int)).

Task 2 – Create table orders with fields – (id (int), date(timestamp)).

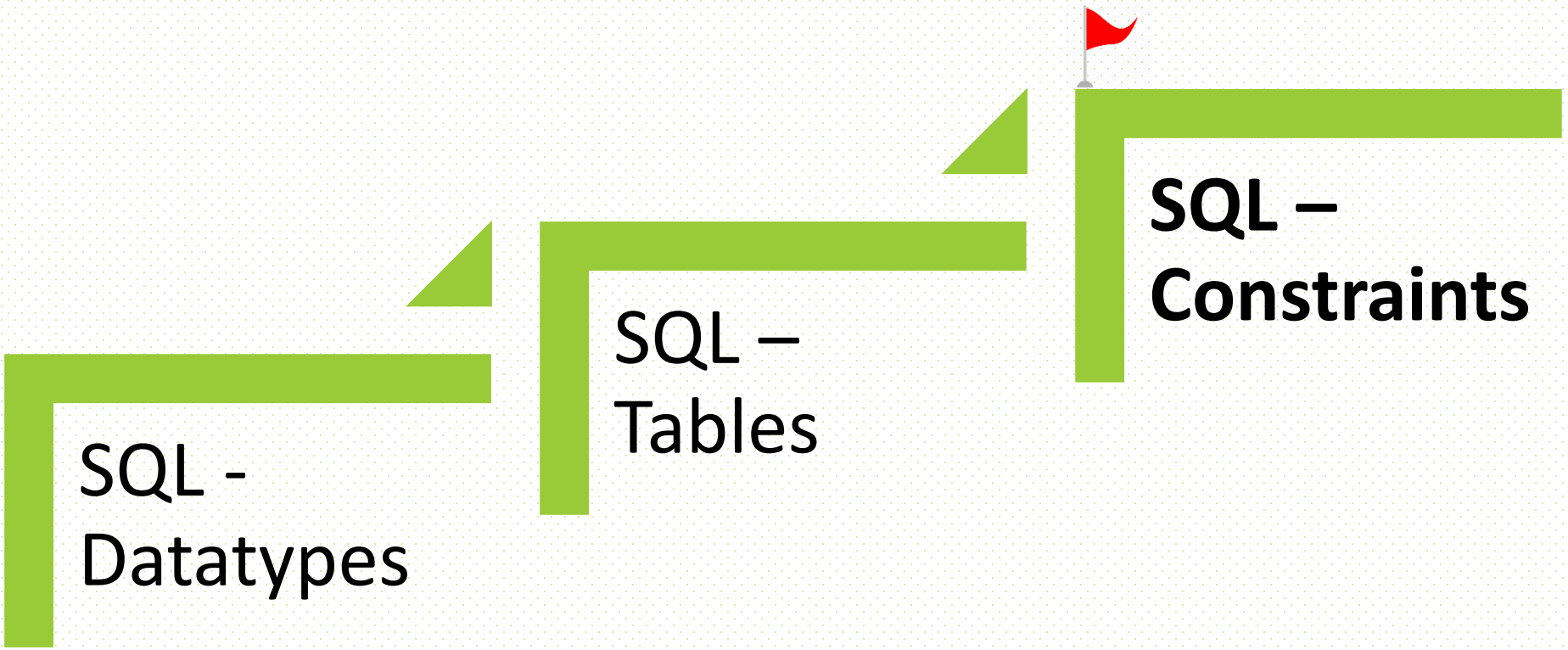
Task 3 – Change type of *customers.pin* to varchar from int.

Task 4 – Add column *username* (varchar) and *email* (varchar) to customers table.

Task 5 – Remove column mobile from customers table.

Task 6 – Drop table orders.

Learning Objectives



SQL – Constraints

- **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 satisfies a specific condition
- **DEFAULT** - Sets a default value for a column when no value is specified
- **INDEX** - Used to create and retrieve data from the database very quickly

SQL Constraints – NOT NULL

Syntax

```
CREATE TABLE <table-name> (
   <column-name> datatype NOT NULL,
);
```

- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.
- This enforces a field to always contain a value, which means that you cannot insert a new record, or update a record without adding a value to this field.
- **Tip:** If the table has already been created, you can add a NOT NULL constraint to a column with the **ALTER TABLE** statement.

SQL Constraints – PRIMARY KEY

Syntax

```
CREATE TABLE <table-name> (
    <column-name> datatype NOT NULL,
    PRIMARY KEY (<column-name> )
);
```

```
ALTER TABLE  <table-name>
ADD PRIMARY KEY (<column-name> );

ALTER TABLE  <table-name>
ADD CONSTRAINT <constraint-name> PRIMARY KEY (<column-name> , <column-name> ,…);
```

```
ALTER TABLE  <table-name>
DROP PRIMARY KEY;
```

SQL Constraints – UNIQUE

Syntax

```
CREATE TABLE <table-name> (
    <column-name> datatype NOT NULL,
    UNIQUE (<column-name> )
    CONSTRAINT <constraint-name> UNIQUE (<column-name> )
);
```

```
ALTER TABLE  <table-name>
ADD UNIQUE (<column-name> );

ALTER TABLE  <table-name>
ADD CONSTRAINT <constraint-name> UNIQUE (<column-name> , <column-name> ,…);
```

```
ALTER TABLE  <table-name>
DROP UNIQUE <constraint-name>;
```

SQL Constraints – FOREIGN KEY

Syntax

```
CREATE TABLE <table-name> (
 <column-name> datatype NOT NULL AUTO_INCREMENT
);
```

```
ALTER TABLE  <table-name> AUTO_INCREMENT=100;
```

SQL Constraints – FOREIGN KEY

Syntax

```
CREATE TABLE <table-name> (
 <column-name> datatype NOT NULL AUTO_INCREMENT
FOREIGN KEY (<column-name> ) REFERENCES <another-table>(<column-name> )
);
```

```
ALTER TABLE  <table-name>
 ADD FOREIGN KEY (<column-name> ) REFERENCES <another-table>(<column-name> )
```

SQL - Tasks

Task 1 – Create table orders with fields – (id (int) <PK, AI> , date (timestamp), customer_id (int) <FK, U>)

Task 2 – Add primary key, auto_increment, unique and not null to customer table using alter table command.

Task 3 – Create table shippers with fields – (id (int) <PK, AI> , name (varchar) <U>, contact (int) <N>)

Task 4 – Add FK constraint in orders table (shipper_id).

Task 5 – Create tables : OrderItems, Products, Suppliers, Categories.

Task 6 – Practice Drop Constraint.