Introduction to Database System

What is a Database

- Database: A database is an organized collection of data, generally stored and accessed electronically from a computer system. ("Database", *Wikipedia*)
- It is a place to store DATA!

Ancient Database

Census file,
Papyrus BM 10068,
late New Kingdom
(~1000 BC),
British Museum,
London

Ancient Database: Egypt Census

- Officials in villages collect and record data on a persistent media (papyrus)
- Media is delivered to and stored in a national archive, and organized for search
- Officials in the national archive will perform statistics on the data and report the result to pharaohs.

Modern Database

IBM 7090 Data Processing System, December, 1959, Source: IBM Archive

Modern Database: US Census

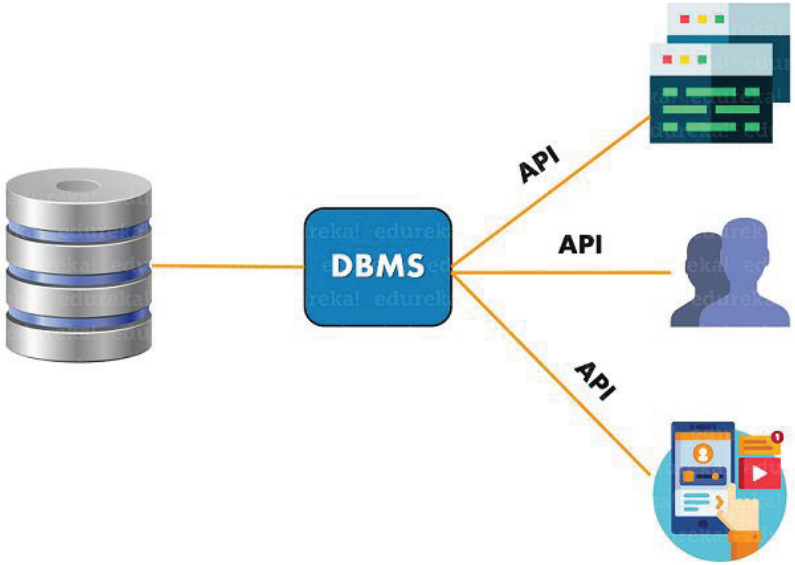
- Officials in each state collect mail-in form and record data on a persistent media (punchcard)
- Media is delivered to and stored in a computer archive, and organized for search
- Officials in federal government will perform statistics on the data and report the result to federal government.

Database System

- Data is stored in persistent electronic media, and organized for retrieval. This storage is called database.
- Storage is managed by a computer application
  - This application that manages database is called a “database management system” (DBMS).
- Different clients connect to DBMS for data access and management
- The whole ecosystem of database, database management system, and clients, is called “database system”

Database System Architecture

- Database is stored on computer hardware (usually hard drive(s))
- DBMS manages data and user connections
- Different clients access database contents by connecting to DBMS



https://www.edureka.co/blog/what-is-dbms

DBMS Definition by Oracle

A DBMS serves as an interface between the database and its end users or programs, allowing users to retrieve, update, and manage how the information is organized and optimized. A DBMS also facilitates oversight and control of databases, enabling a variety of administrative operations such as performance monitoring, tuning, and backup and recovery.

https://www.oracle.com/database/what-is-database/

DBMS and Clients

- Database and DBMS usually reside on a server (or multiple servers) due to the volume of data
- Users access data by connecting to DBMS from a ***client software***
  - Data accessing software (browser based or application)
    - Example: pgAdmin, DBeaver
  - Computer scripts/programs using data: python and/or Java

PostgreSQL Introduction And Installation

PostgreSQL Architecture

- Concepts involved
  - Database storage: On your local machine's hard drive
  - DBMS: PostgreSQL service
  - Client Application: pgAdmin
  - User: You

Compatibility

- There are many RDBMS out there
  - Each has its own SQL dialect
- There is also a SQL standard (latest in 2016)
  - Not always followed
- In general, 95% of SQL are compatible (the general SQL statement)
  - Be prepared for the remaining 5%: usually data type definitions and procedure/functions

Installing PostgreSQL

- Tasks:
  - Installing PostgreSQL (DBMS and pgAdmin)
    - During installation, PostgreSQL will create a default database on your machine
  - Use pgAdmin to connect to the local host DBMS and access the default database
  - Running sample SQL to confirm the installation

Introduction to Relational Database

Early Database Implementation

- The simplest database can be just one file, e.g., Excel spreadsheet.
- The challenge is how to present large datasets in computer.
  - Modern data is complex and large in size
  - Must use formal design and modeling techniques
- Before 1970s, use file system for data storage:
  - No clear relationships between files. Hard to manage.
  - Data duplication/loss and security concerns
- Some experimental database designs: network, hierarchy, e.g.

Relational Database

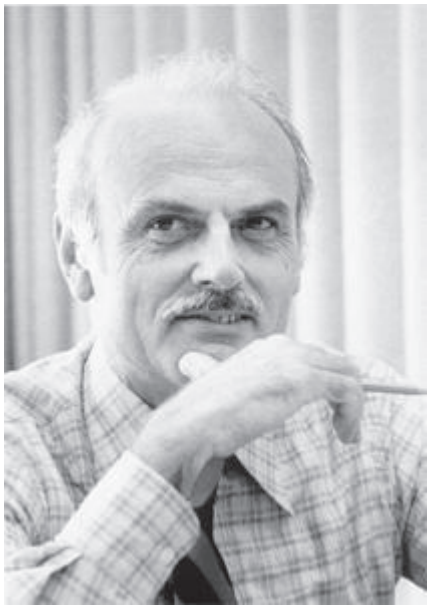
- In 1970, Dr. Codd from IBM proposed Relational Model, a better way of database design
  - Database that is based on Relational Model is called relational database

Relational Database

- Quickly gain momentum and became mainstream
- Most commercial or open source database system today are relational
  - Commercial: Oracle, SQL Server
  - Open source: MySQL, PostgreSQL
- There are other, non-relational databases, but relational database is the main stream and is the main topic of this course.

Relational Model

- Relational Model by Dr. Edgar F. Codd (1970):
  a. Even the most complex datasets can be represented as ***Relations***.
  b. A relation is a ***set*** of ***tuples***. Tuples contains manageable information.
  c. Relations are operated through Relational Algebra.



Relational Model Example

- Relation: Course
- Tuple (defined sequence of elements):

  DS530, Big Data and Data Management, 3 Credits, Wed 6-10pm

  DS620, Data Visualization, 3 Credits, Tue 6-10pm

Attribute

- Attribute: Property of each element in the tuple

  Code, Name, Credit, Time

  DS530, Big Data and Data Management, 3 Credits, Wed 6-10pm

  - All tuples in a relation have same attributes, in the same order.
  - Each attribute has its own data type: for better processing of data

Relation

- Each relation represents a real world “thing”: Will cover in detail later
- Each tuple represents a real world instance of that “thing”
- Each attribute represents a characteristics of that instance

Defining a relation = describing the attributes of the relation.

Populating a relation = inserting instance into the relation

Attribute Restraints

- Attribute may have restraints
  - Some attributes are
    - required: a value must be present for each tuple
    - optional: No value is required (home phone number, e.g.)
  - Some attributes are
    - Unique: Each tuple has a different value
  - Some attributes have boundaries
    - Month: 01-12. Weekday 1-7 (or 0-6)
    - Sales (no return): >0

Attribute Data Type

- Common Data Types
  - Text, String, Characters: Course Code and Name
  - Integer/Decimal: Credit (aggregable)
  - Date/Time: Course date/time

Composite Attribute

- Some attributes can to put together to form a more meaningful attribute
  - Year + Month + Day => Date
  - Building + Room Number => Classroom ID
- This is called “composite attribute”

Shorthand Representation

- Shorthand representation

  Relation (Attribute 1, Attribute 2, Attribute 3, ...)

- Example:

  Student (Name, ID, DOB, Address)
  Course (ID, Name, Department, Location)

- Qualified Representation: Add relation name to qualify the attribute

  Student.Name vs Course.Name

Relation as a 2D table

- Relation consists of Tuples. All tuples in a relation have same attributes, in the same order. As such, a relation can be simply displayed as a 2 dimensional table

Code	Name	Credit	Date/Time
DS530	Big Data and Data Management	3 Credits	Wed 6-10pm
DS620	Data Visualization	3 Credits	Tue 6-10pm

Functional Dependency and Primary Key

Uniqueness of Tuple

- Each tuple in the set should be unique
- This is the natural result of proper database design
  - Real world instances are unique
  - Tuples in relation, as mappings of real world instances, should also be unique
- How to identify each tuple?
  - What determines the attribute value of each tuple?

Functional Dependency

- Relationships between attributes
- Y is functional dependent on X ($X \rightarrow Y$) if and only if each X value is associated with precisely one Y value. ("Functional Dependency", *Wikipedia*)
  - Knowing X's value, you can immediately know Y's value: ID -> Name
- Functional Determination: The other way of expressing functional dependency

Composite Determination

- Functional Dependency can happen between sets of attributes

  State + Car Plate -> Car Make, Car Model, Car Color, etc

  NJ 1234567 -> Ford, Mustang, Red, etc.

  Eventually, a set of attributes can functional determine all the attributes of the relation.

Candidate Key

- If a set of attributes can functionally determines all attributes in a relation, this set of attributes "functionally determines" the relation.
  - We call this set of attributes the <u>key</u> of this relation, e.g., Student ID
    - Can have multiple attributes (State + Car Plate)
- Candidate key is a set of attributes such that:
  - 1: The relation is functionally dependent on this set of attributes.
  - 2: There is no subset of the attributes for which (1) holds.

Candidate Key

- Candidate key is a set of attributes such that:
  - 1: The relation is functionally dependent on this set of attributes.
  - 2: There is no subset of the attributes for which (1) holds.

State + Car Plate -> Car

State + Car Plate + Car Make -> Car

Subset of 2nd set

Primary Key

- Primary key = best candidate key
  - Conceptual consideration: which looks more natural?
  - Practical consideration: Integer/Datetime provides best performance. Varchar performs worst.
    - ID *vs* SSN *vs* Name + Address
  - Other considerations:
    - SSN as primary key (privacy issue): HICN vs MBI

Shorthand Representation

- Shorthand representation of primary key: Use <u>underscore</u> or *...* to enclose each attributes of the primary key.

  Student (<u>ID</u>, Name, DOB, Address)
  Course (<u>ID</u>, Name, Department, Location)
  or
  Student (*ID*, Name, DOB, Address)
  Course (*ID*, Name, Department, Location)

Introduction to SQL

Structured Query Language (SQL)

- ***THE*** standard way of relational database operation
  - Create/Maintain/Drop tables
  - Insert/Update/Delete/Query table data
- Descriptive, not procedural
- SQL has an ANSI Standard: Same across different vendors
  - Latest is SQL-2016

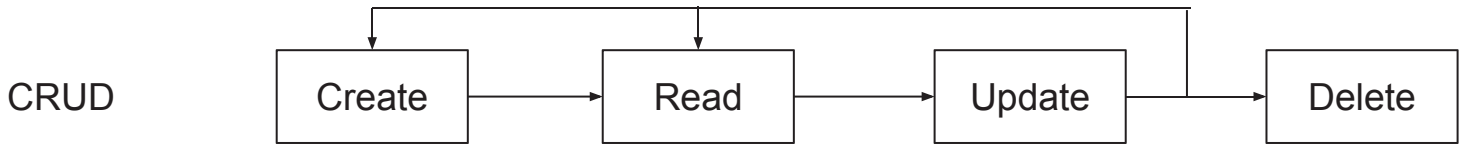
Overview of SQL

- 2 types of statements
  - Data Definition Language (DDL): CREATE, DROP, ALTER
  - Data Manipulation Language (DML): INSERT, UPDATE, DELETE, SELECT
- 7 most important statements of SQL
  - CREATE, ALTER, INSERT, SELECT, UPDATE, DELETE, DROP

Data Lifecycle

How data exists in computer systems: Create, Read, Update, Delete (CRUD)

- Generic lifecycle of data in ANY computer system, not SQL specific.

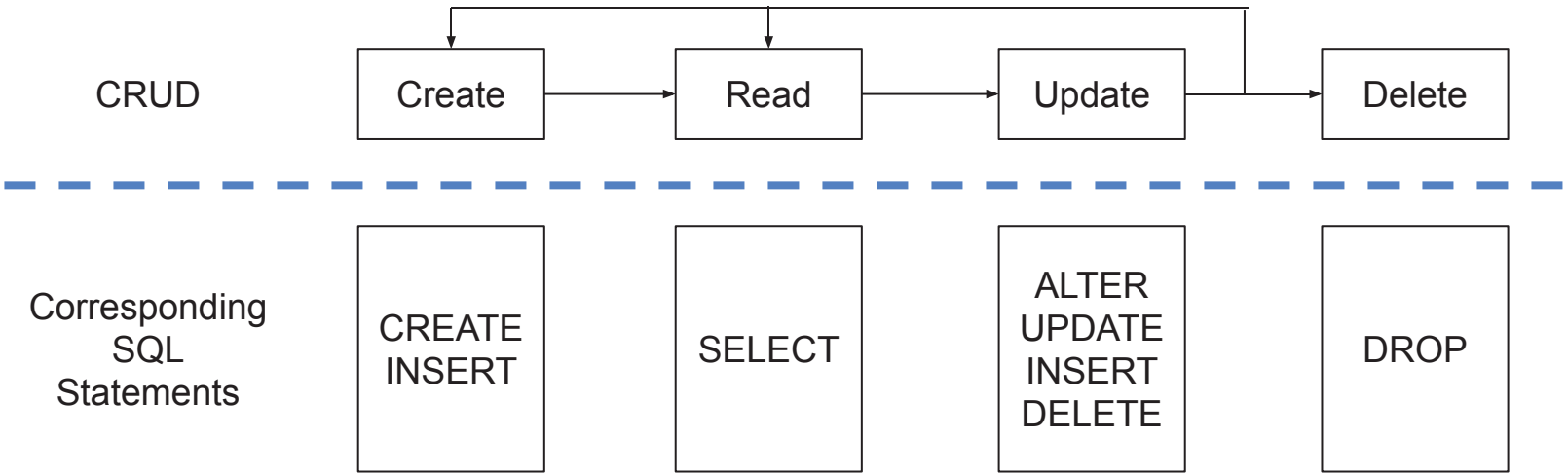


CRUD is achieved by using SQL in relational databases.

Data Lifecycle

How data exists in computer systems: Create, Read, Update, Delete (CRUD)

- Generic lifecycle of data in ANY computer system, not SQL specific.

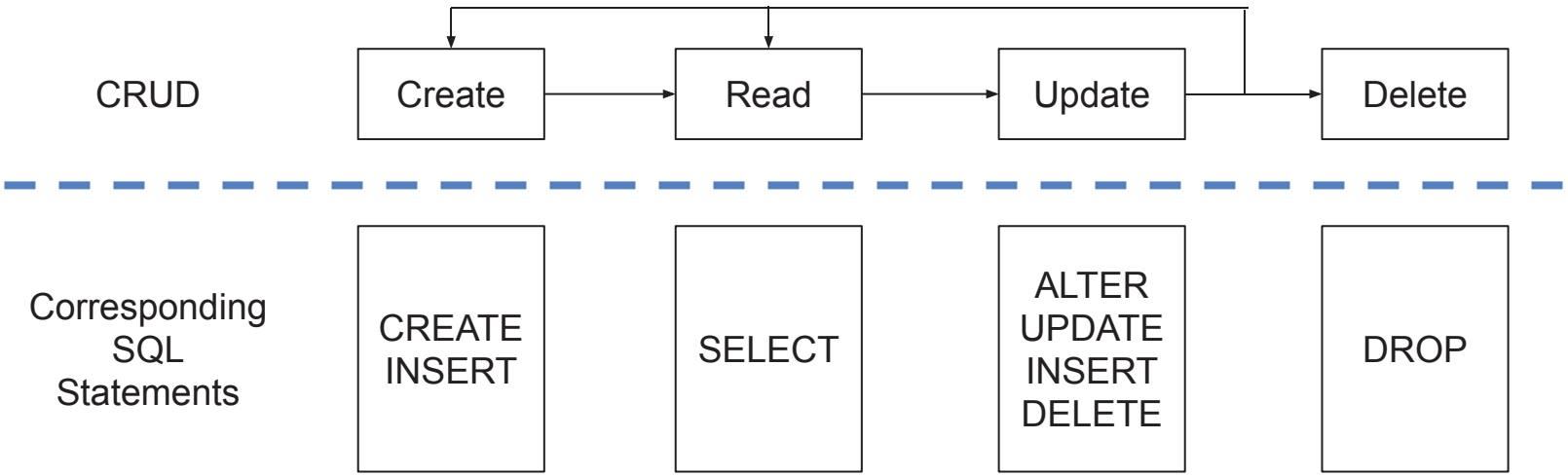


CREATE and DROP

Data Lifecycle

How data exists in computer systems: Create, Read, Update, Delete (CRUD)

- Generic lifecycle of data in ANY computer system, not SQL specific.



CREATE DATABASE Statement

- CREATE DATABASE statement: Creates a database.

```
CREATE DATABASE Academy;
```

Table Creation

- Each table represents a “relation”
- Each row represents a tuple, which is a real world instance
- Each column represents an attribute

Defining a table = describing the columns (attributes) of the table.

Populating a table = inserting row (instance) into the table

CREATE TABLE Statement

- CREATE statement

Instructor (Instructor Name, Instructor Affiliation)

```
CREATE TABLE Instructor (
    Instructor_Name CHAR(100) PRIMARY KEY,
    Instructor_Affiliation CHAR(10)
)
```

CREATE Statement Syntax

- Statement keyword: `CREATE TABLE`
- Table name must be defined: `<table_name>`
- Table content inside parenthesis `()`
- Column definitions are listed inside parenthesis

```
CREATE TABLE Instructor (
    Instructor_Name CHAR(100) PRIMARY KEY,
    Instructor_Affiliation CHAR(10)
)
```

Naming Standard

- Table and column names
  - <= 31 characters
  - Start with a letter or an underscore (_)
  - Can contain letter, number, and underscore (_)
  - No space, No special characters.
- Quoted name (case sensitive):

  “2022-sales”

Valid Names

- Valid names:

  Student_ID
  Teacher_Name
  StudentID
  DS530_Reservation
  MyName____
  _Good_Name_in_PostgreSQL

Invalid Names

- Invalid Names

  Student ID
  Teacher_Salary_$
  2022_Sales_Goal
  This_Column_Name_Is_Too_Long_In_Most_Relational_DBMS

- Add double quotes to make these valid names

CREATE

```
CREATE TABLE <table_name> (
<column_1_definition>...
)
```

- Use relation name as `<table_name>`
- Use attribute name as column name. Each attribute is one column.
- Add data type to each column
- Add comma to the end of all definitions except for the last one.

Table Content Definition

- Column definitions first

  ```
  <column_name> <data_type> [<column_property>]
  ```

- Followed by other table properties: PRIMARY KEY
- Enclosed in parenthesis.
- Separated by comma (no comma for last content).

```
CREATE TABLE Instructor (
    Instructor_Name CHAR(100),
    Instructor_Affiliation CHAR(10),
    PRIMARY KEY (Instructor_Name)
)
```

Column Definition

- Column definition is determined by the attribute definition

```
<column_1_definition> =
<column_name> <data_type> <column_constraints>,
```

Constraints

- Depending on the relation/attribute definition
- Relation
  - Primary Key
- Column
  - Unique
- Defined in CREATE statement, or appended to table after CREATE statement

Column Constraint

- Column Constraints
  - UNIQUE

    ```
    Instructor_SSN CHAR(9) UNIQUE,
    ```

  - DEFAULT value

    ```
    Course_Credit_Hour INTEGER DEFAULT 3,
    ```

  - More...

PRIMARY KEY as Column Constraint

- Can be specified either at table property section or at column property section:
  - Single Column Primary Key: After column data type, before comma

```
CREATE TABLE Instructor (
    Instructor_Name CHAR(100) PRIMARY KEY,
    Instructor_Affiliation CHAR(10)
)
```

PRIMARY KEY as Table Constraint

- Compound Primary Key: Table constraint

```
CREATE TABLE Instructor (
    Instructor_Name CHAR(100),
    Affiliation CHAR(10),
    PRIMARY KEY (Instructor_Name, Affiliation)
)
```

DROP TABLE Statement

- DROP TABLE statement

  ```
  DROP TABLE <table_name>
  ```

  Drop one table at a time.

DROP DATABASE Statement

- DROP DATABASE statement

```
DROP DATABASE <Database_name>
```

Data Types

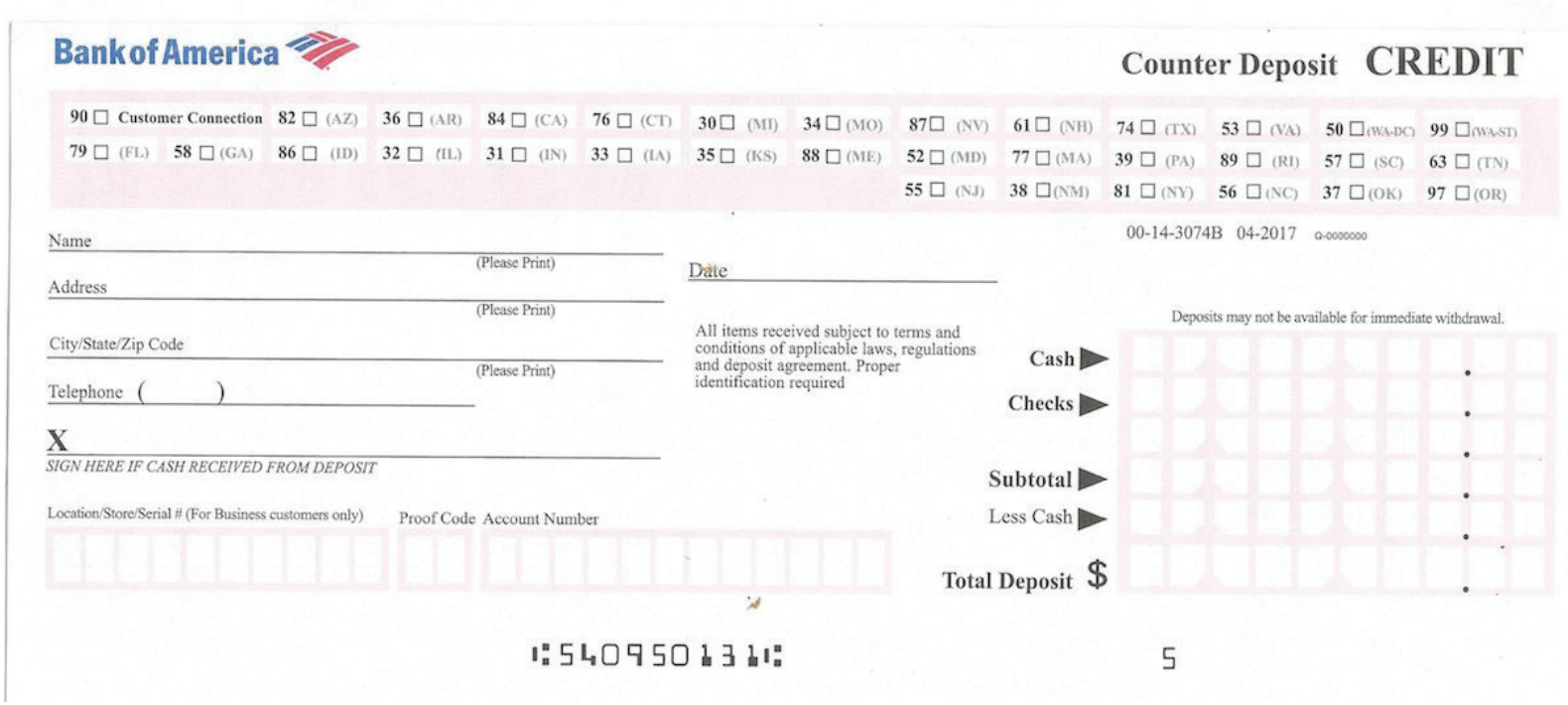
Column Data Types

- Defines what value the column can hold, and what operations can be performed.
  - Example: Sales
  - Example: Course Name
- Defines how this column is stored, what is the size of the column.

Common Data Types

- Common Data Types
  - Integer/Decimal
  - Text, String, Characters
  - Date/Time

Common Data Types



Bank of America

Counter Deposit CREDIT

90 ☐ Customer Connection 82 ☐ (AZ) 36 ☐ (AR) 84 ☐ (CA) 76 ☐ (CT) 30 ☐ (MI) 34 ☐ (MO) 87 ☐ (NV) 61 ☐ (NH) 74 ☐ (TX) 53 ☐ (VA) 50 ☐ (WA-DC) 99 ☐ (WA-ST)
79 ☐ (FL) 58 ☐ (GA) 86 ☐ (ID) 32 ☐ (IL) 31 ☐ (IN) 33 ☐ (IA) 35 ☐ (KS) 88 ☐ (ME) 52 ☐ (MD) 77 ☐ (MA) 39 ☐ (PA) 89 ☐ (RI) 57 ☐ (SC) 63 ☐ (TN)
55 ☐ (NJ) 38 ☐ (NM) 81 ☐ (NY) 56 ☐ (NC) 37 ☐ (OK) 97 ☐ (OR)

00-14-3074B 04-2017 0-0000000

Name (Please Print)

Address (Please Print)

City/State/Zip Code (Please Print)

Telephone ( )

X

SIGN HERE IF CASH RECEIVED FROM DEPOSIT

Location/Store/Serial # (For Business customers only) Proof Code Account Number

Date

All items received subject to terms and conditions of applicable laws, regulations and deposit agreement. Proper identification required

Deposits may not be available for immediate withdrawal.

Cash ▶

Checks ▶

Subtotal ▶

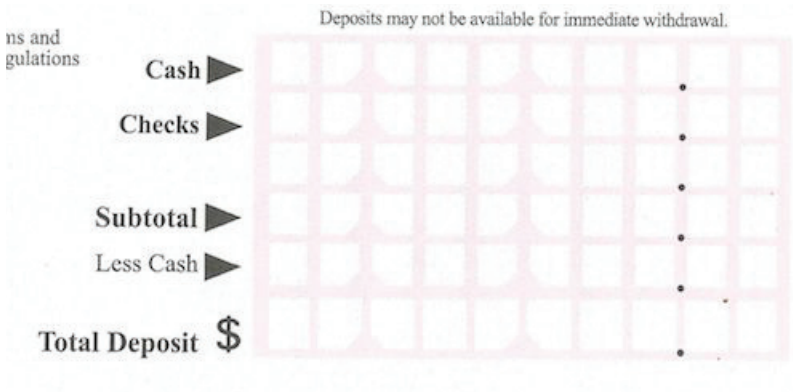
Less Cash ▶

Total Deposit $

⑆540950131⑆ 5

Decimal

Numbers of fixed width, both for integer part and decimal part

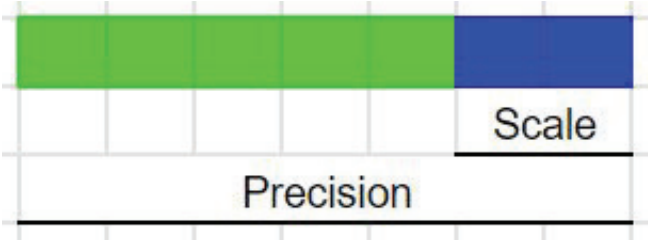


Decimal

- DECIMAL(precision,scale)

  DECIMAL(7,2)

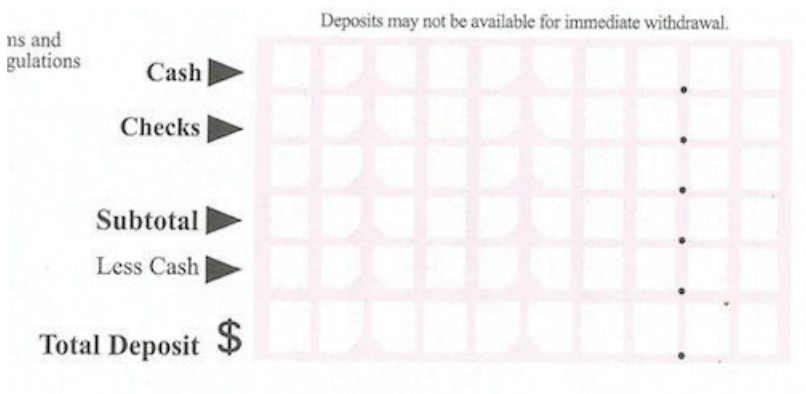
  - Integer when scale = 0




```
Sales_Amount DECIMAL(7, 2),
Number_Of_Customers DECIMAL(5, 0),
```

Decimal

- Common usages:
  - Currency (usually with Scale = 2)
  - Percentage



Integer/INT

- Decimal with width of 0: so common that we created a data type for it
- Integer number: Quantity of items sold, Number of days
  - Regularly used as primary key because of low storage cost and good performance

```
Instructor_ID INTEGER,
Instructor_AGE INT,
```

Common Mistakes for DECIMAL

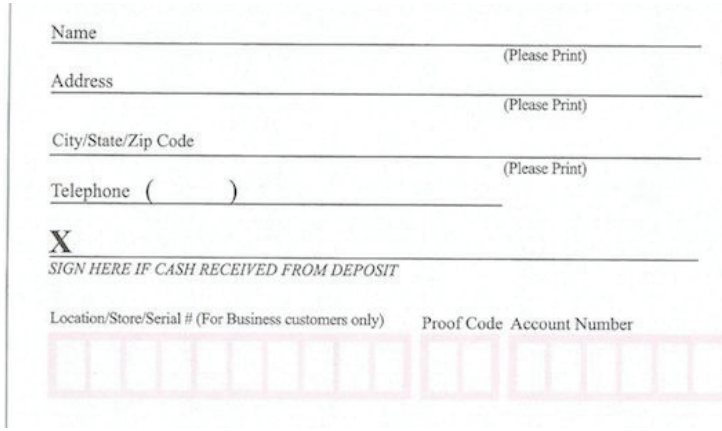
- Place inside quote: “123.45”
- Comma and currency sign: 12,345.67 $123.45
- Overflow when number of whole number digits > Precision - Scale: Insert 123456.78 into DECIMAL(7,2)
- Lost of accuracy when number of decimal digits > Scale: Insert 123456.78 into DECIMAL(7,0)
    - Note: This will not result in a SQL execution failure

Common Mistakes for Integer

- Place inside quote: ‘1234’
- Comma: 12,345
- Decimal points

Text/String/Characters

- Two types of texts
  - Fixed length: Account ID, Zip, State Code, License Plate
    - CHAR
  - Varying length: Name, Address
    - VARCHAR


Name
(Please Print)
Address
(Please Print)
City/State/Zip Code
(Please Print)
Telephone ( )
X
SIGN HERE IF CASH RECEIVED FROM DEPOSIT
Location/Store/Serial # (For Business customers only)
Proof Code
Account Number

CHAR vs VARCHAR

- CHAR has a fixed length, while VARCHAR has a varying length, up to the defined scope.
- Fix-length columns (car license plate, driver ID) should use CHAR. Variable length columns (name, address) should use VARCHAR.
- Should SSN/Phone Number/Zip Code be CHAR or INT?
    - General guideline is that if a column cannot or should not be summed up (What does SUM of SSN mean?), it should be text (CHAR).

Text/String/Characters

- Text is usually defined with a size (length)

```
Course_Code CHAR(5),
Course_Name VARCHAR(100),
```

- CHAR(n) and VARCHAR(n) difference: Database will fill in spaces in the rest of characters
  - CHAR(10) with ‘Jedi’ => ‘Jedi ’ --- 6 extra spaces
  - VARCHAR(10) with ‘Jedi’: No fill

Text Values

- Single quote delimited string: 'city', 'City', 'CITY'
  - Single quote is used to delimit the string. It is not a part of string content.
- Case sensitive inside quotes
- Use two single quotes to represent single quote:

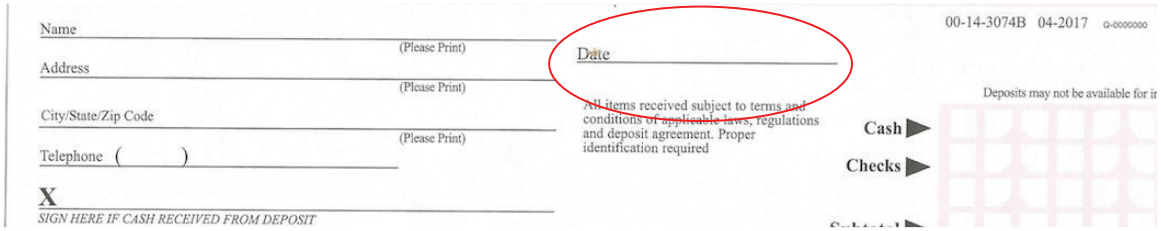
  I'm Happy => 'I''m Happy'

Common Mistakes of Text

- Forget single quote: Opposite to DECIMAL/INTEGER
- Use double quote as delimiter: “City” is not a proper text string
- Use wrong character of quotes when copying from rich format text editor (such as Word or Outlook): Must be plain text single quote for SQL to recognize it as delimiter

‘City’ vs 'City'

Date/Time



- DATE: '07/04/2022'
- TIME: '04:45:59.135', '04:45 am'
- TIMESTAMP: Both the date part and the time part

  '07/04/2022 04:45:59'

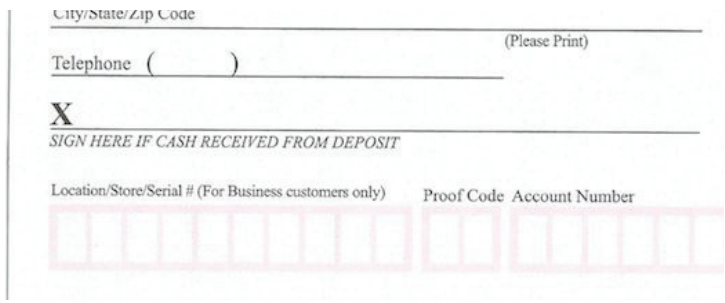
  Use single quote around the value for all three types.

Date/Time Format

- Date/Time/Timestamp data are stored in standard format in database, utilized/displayed based on your client setting.
  - Time Zone: EST, PST, etc.
  - Date Format: 07/14/2022, 14/07/2022, 2022-07-14
  - Usually goes with your local machine's setting: US EST setup
  - Too complex to summarize in class. Always test several scenarios when you set up your client.

Binary/File

- Binary (video, audio, image, etc.): Signature
  - Store the binary in database
  - Store in a file system and save the file address in database



Default Size for Data Types

- Default sizes:

  CHAR = CHAR(1)

  VARCHAR: Up to the implementation limit (PostgreSQL)

  DECIMAL: Up to the implementation limit (PostgreSQL)

Default Size for Data Types

- Specify the size even though you don't have to:
  - Not portable:
    - DECIMAL definition different from SQL standard
    - Different DBMS may have different implementations
  - Confusion and wrong assumption: CHAR vs VARCHAR

SQL Demo

SQL Convention in PostgreSQL

- Running multiple statements
  - Highlighting
  - Semicolon (;)

SQL Convention in PostgreSQL

- SQL Statements
  - Keywords and table/column names are not case sensitive.
  - Line breaks and extra spaces are ignored.
  - But still, try to form a convention
    - Using upper case for keywords).
    - Proper indentation

Online SQL Practice for Week 1

- Use the following sight for first week's CREATE/DROP practice, before you install PostgreSQL on your own machine.

  http://sqlfiddle.com/

  Practice CREATE and DROP