Introduction to Back End Testing

The latest in smart home security

ring

Ring Smart Lighting Spotlight

Find a perfect gift

Unique gift ideas from Gift Finder

See more

Ava's picks

Shop Ava's dorm picks.

Shop College Dorm Essentials

5 days to go

A two-day parade of epic deals starts July 15, 12AM PT.

Shop early deals

Prime Day is July 15 & 16

Get ready with tips and how-to's so you can shop like a pro.

Read the Prime Day Guide

Happy School Year

Shop everything for a great year

Front-end Testing Tools

Cybertek

Back End Testing Tools

WHAT IS DATA?

- Piece of information
- For example Bank account ?
  - Account Number ->123
  - Account Type -> Checking
  - User Firstname -> John
  - Last name -> Smith
  - Balance ->100,000

WHAT IS DATA ?

- All above data needs to be stored somewhere, where **it is secure**, **easy to ready, fast to read, easy and fast to update.**
- In **databases** we store **data** in an organized manner.

Cybertek

WHAT IS DATABASE ?

- Database is a systematic collection of data.
- Databases support storage and manipulation of data.
- Databases make data management easy

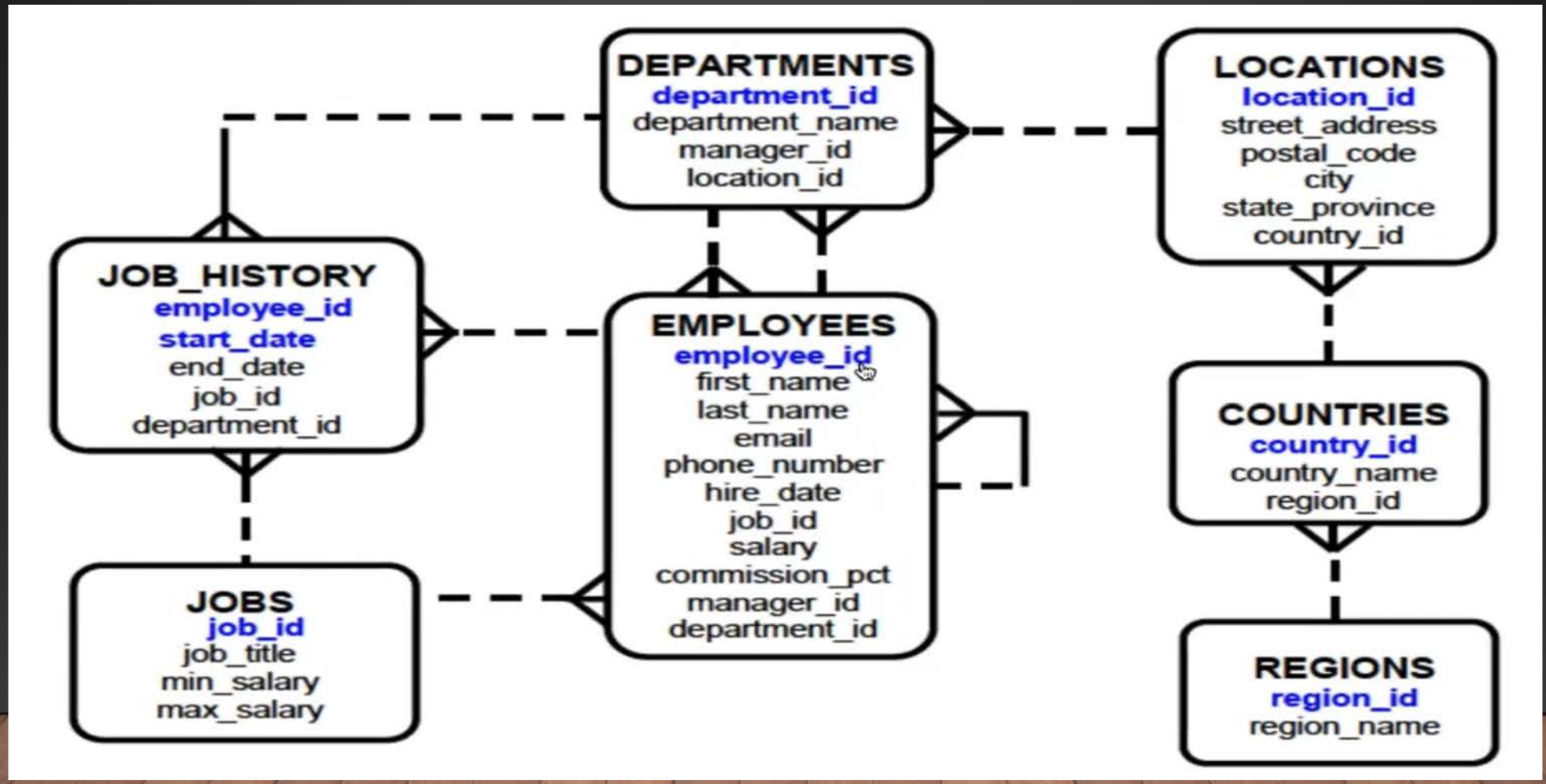
Relational DATABASE

- organize that data in a series of tables

Employee_ID	Employee_Name	Employee_Address	Employee_Salary
#007	John	VA	95k
#008	James	KY	100k
#009	Aaron	CA	105k
#010	Luckus	WA	110k

Relational DATABASE

- tables are related to each other using Primary and foreign keys



Non Relational DATABASE

- All Data are in Key & Value format

```
{
   first_name: 'Dexter',
   last_name: 'Lanas'
   city: 'Vancouver'
   location: [45.123,47.232],
   phones: [
      { phone_number: '111-111-1111',
        type: mobile,
        person_id: 1, ... },
      { phone_number: '444-444-4444',
        type: home,
        person_id: 1, ... },
      { phone_number: '777-777-7777',
        type: office,
        person_id: 1, ... },
   ]
}
```

Database Management System

- Create, maintains and organize the database
- Easier to manage the database
- Improves the availability of the information
- Does have backup system

RDBMS

- RDBMS --> Relational Database management system.
- All RDBMS using SQL language
- Relational Database --> tables are related to each other using Primary and foreign keys

WHAT IS SQL ?

- SQL - > STRUCTURED QUERY LANGUAGE
- SQL is a language that is used to work with Databases and manipulate data.

SQL is a hybrid language

- SQL is combined with four languages:
  - Data Query Language(DQL)
  - Data Definition Language (DDL)
  - Data Control Language (DCL)
  - Data Manipulation Language (DML):

What is Query in SQL?

- A set of instructions
- Telling Database Management System that what we would like to do.

```
1
2
3
4   select * from Employees;
5
6   select email from Employees where first_name='Steven' and last_name='King';
7
8
```

Data Types in Query

- Int & Integer: whole numbers
- Decimal: decimal numbers
- Varchar : String of text
- Date: 'YYYY-MM-DD'
- Timestamp: 'YYYY-MM-DD HH:MM:SS' -
- Boolean: true & false, Boolean expressions

SELECT STATEMENT

- First, we specify a list of columns in the table from which we want to query data in the **SELECT** statement. We use a comma between each column in case we want to query data from multiple columns.
- If we want to query data from all column, we can use an **asterisk (*)** as the shorthand for all columns.
- Second, we indicate the **table name** after the FROM keyword
- SQL language is case INSENSITIVE

Cybertek

SELECT STATEMENT

- The following illustrates the syntax of the SELECT statement:

```
SELECT column1, column2... FROM table name ;
```

keyword — keyword

Cybertek

SELECT STATEMENT SYNTAX

- Select * From TableName;
- Select ColumnName From TableName;
- Select ColumName1, ColumName2 ... From TableName;
- Select Column(s) From TbaleName1, TableName2 ;

Cybertek

SELECT DISTINCT STATEMENT

- The **DISTINCT** keyword can be used to return only distinct (different) values.

```
SELECT DISTINCT column1, column2... FROM table_name ;
```

Removes duplicate values

Cybertek

SELECT WHERE STATEMENT

- The WHERE clause appears right after the FROM clause of the SELECT statement.
- The conditions are used to filter the rows returned from the SELECT statement.
- PostgreSQL provides us with various standard operators to construct the conditions.

WHERE CLUASE SYNTAX

SELECT *column_1, column_2.. column_n*

FROM *table_name*

WHERE *conditions*;

Applies filter to result

Cybertek

WHERE STATEMENT OPERATORS

SELECT WHERE Statement

OPERATOR	DESCRIPTION
=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<> or !=	Not equal
AND	Logical operator AND
OR	Logical operator OR

Cybertek

COUNT STATEMENT

- The COUNT function returns the number of input rows that match a specific condition of a query.

```
SELECT COUNT(*)  FROM tablename;
```

Cybertek

COUNT STATEMENT

- The COUNT(*) function returns the number of rows returned by a SELECT clause.

  How many departments do we have ?

COUNT STATEMENT

- Similar to the COUNT(*) function, the COUNT(column) function returns the number of rows returned by a SELECT clause.
- However, it does not consider NULL values in the column.

COUNT STATEMENT

- We can also use COUNT with DISTINCT, for example;

  How many different type of amount we have ?

  How many different rating type we have ?

ORDER BY STATEMENT

The ORDER BY clause allows you to sort the rows returned from the SELECT statement in ascending or descending order based on criteria specified.

ORDER BY STATEMENT

- The following illustrates the syntax of the SELECT statement:

```
SELECT column 1, column 2

FROM table name

ORDER BY column_1 ASC / DESC;
```

The column name we want order by

Which order we want ASC or DESC

Cybertek

BETWEEN STATEMENT

- WE use the BETWEEN operator to match a value against a range of values. For example;

Value BETWEEN low AND high

BETWEEN STATEMENT

- If the value is greater than or equal to the low value and less than or equal to the high value, the expression returns true, or vice versa.
- We can rewrite the BETWEEN operator by using the greater than or equal (>=) or less than or equal (<=) operators as the following statement:

```
value >= low AND value <= high
```

```
value BETWEEN low AND high
```

Same statement

Cybertek

IN STATEMENT

- We use the IN operator with the WHERE clause to check if a value matches any value in a list of values.
- The syntax of the IN operator is as follows:

value **IN (valuel,value2,…)**

Cybertek

IN STATEMENT

- The list of values is not limited to a list of numbers or strings but also a result set of a SELECT statement as shown in the following query:

```
Value IN (SELECT value FROM tbl_name)
```

- Just like with BETWEEN, you can use NOT to adjust an IN statement (NOT IN)

LIKE

- Suppose the store manager asks you find an employee that he does not remember the name exactly.
- He just remembers that employee's first name begins with something like Jen.
- How do you find the exact employee that the store manager is asking?

LIKE STATEMENT

- You may find the employee in the employee table by looking at the first name column to see if there is any value that begins with Jen.
- It is kind of tedious because there many rows in the customer table.

LIKE STATEMENT

- Fortunately, we can use the LIKE operator to as the following query:

```
SELECT first name, last name
FROM employee
WHERE first name LIKE 'Jen%';
```

Single Quotes

% = pattern matching(take whatever after Jen)

Cybertek

LIKE STATEMENT

- The query returns rows whose values in the first name column begin with Jen and may be followed by any sequence of characters.
- This technique is called pattern matching.

LIKE STATEMENT

- You construct a pattern by combining a string with wildcard characters and use the LIKE or NOT LIKE operator to find the matches.
  - Percent (%) for matching any sequence of characters.
  - Underscore (_) for matching any single character.

Cybertek

AGGREGATE FUNCTIONS

1. MIN
2. MAX
3. AVG
4. SUM

MIN

- Performs the action for multiple rows at once and returns single result
- Checks all the rows and shows minimum one.

```
SELECT MIN(salary)
FROM employees;
```

Column Name

Cybertek

MAX

- Checks all the rows and shows maximum one.

```
SELECT MAX(salary)
FROM employees;
```

AVG

- Add all rows and get average.

```
SELECT AVG(salary)
FROM employees;
```

ROUND

- ROUND the result with given decimal.

```
SELECT ROUND(AVG(salary), 2)
FROM employees;
```

How many decimals we want to see

Cybertek

SUM

- Add all rows and shows SUM.

```
SELECT SUM(salary)
FROM employees;
```