**1.  What is software?**

Software is a collection of instructions (class), code, programs that enable the user to interact with a computer, its hardware, or perform tasks. For example, without your Internet browser software, you could not surf the Internet or read this page. Without an operating system, the browser could not run on your computer.

Two types of software: System software & Application Software

Is a process that executes set of instructions and code to achieve a specific functionality.

Ex : addicting two numbers, executing 2 numbers, executing statistics, executing any scientific formulas

**2. What is an object and class in software with own example**

Class is a blueprint for creating objects (a particular data structure)

**3. What is the object oriented programming and sequential programming and their differences and advantages.**

**4. What is hardware?**

Hardware consists of tangible resources such as mice, keyboards, USB ports, CPUs, memory, printers, and so on. Phones are hardware. iPads, Kindles, and Fire TV sticks are hardware. Hardware and software work together to make a system functional.

**5. What is a middleware?**

Middleware is software that is used to bridge the gap between applications and other tools or databases. Middleware sits between an operating system and the applications that run on it.

Middleware appears in many locations; however, organizations and developers make specific use of middleware to more

efficiently build applications

Specially focused only on transferring of data(routing, cleaning, transforming) between source to destination. Handle special instructions.

Combination of software and hardware.

The name middleware stems from the fact that it is the software that sits between the client-side requests on the front end and the back-end resource being requested.

<https://searchapparchitecture.techtarget.com/definition/middleware>

**6. Differences of software/ middleware and hardware**

Hardware is the physical things that you can put your hands on - ie the circuit boards, the physical cases etc.

Software is the programs which run on that hardware;

Software = object and class(technical terms)

Computer hardware is any physical device used in or with your machine, whereas software is a collection of code installed onto your computer's hard drive. For example, the computer monitor you are using to read this text and the mouse you are using to navigate this web page are computer hardware

Middleware is a particular type of software which is designed specifically as glue between other software. In a way middleware is nothing special, it is written in the same way as all other software but instead of having a set of user interfaces, it provides an interface to other software.

**7. What is a web page?**

A web page (also written as webpage) is a specific collection of information provided by a website and displayed to a user in a web browser.

**8. What is web development and how it is different from application development?**

Web development is the building and maintenance of websites; it’s the work that happens behind the scenes to make a website look great, work fast and perform well for the best UX

Web development = front end(Look and feel only)- html, css, javascript, angular js

Application development: core Functionality and outcome of the business, internet connectivity(Java, Dot net)

Back end Development: developing data bases and tables(Oracle Db2, DBA)

Full Stack: Web Development + Application Development + Back end

**Middleware – In between Application and Back end**

**9. What is security?**

T security is a set of cybersecurity strategies that prevents unauthorized access to organizational assets such as computers, networks, and data. It maintains the integrity and confidentiality of sensitive information, blocking the access of sophisticated hackers.

Although IT security and information security sound similar, they do refer to different types of security. Information security refers to the processes and tools designed to protect sensitive business information from invasion, whereas IT security refers to securing digital data, through computer network security.

Authorization (limited access)

Authentication (username and password)

Auditing (Time Logging)

**10. What is governance?**

IT governance (ITG) is defined as the processes that ensure the effective and efficient use of IT in enabling an organization to achieve its goals

Governance = President (Starting from Security to communicating)

Set of administrative policies that have to followed by entire software organization

**11. What is a web service?**

web services are XML-centered data exchange systems that use the internet for A2A (application-to-application) communication and interfacing. These processes involve programs, messages, documents, and/or objects.

The term Web services describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone.

Web services allow different applications from different sources to communicate with each other without time-consuming custom coding, and because all communication is in XML, Web services are not tied to any one operating system or programming language. For example, Java can talk with Perl, Windows applications can talk with UNIX applications.

Any transactions happen on web page in an internet.

**12. What is integration in general?**

Integration is the connection of data, applications, APIs, and devices across your IT organization to be more efficient, productive, and agile.

Integration is a process that connects the various IT systems and applications in an enterprise so that they work cohesively in a coordinated and unified manner. In short, system integration is like putting a puzzle together. There are scattered pieces of an organization’s information subsystems that need to fit together into one well-coordinated, cohesive architecture or integrated application mesh. It’s a complex building process that connects an organization’s functions from varying systems, streamlining disparate systems, including existing hardware, software (customized or out-of-box), and communications.

**13. What is service oriented architecture**

A service-oriented architecture (SOA) is an architectural pattern in computer software design in which application components provide services to other components via a communications protocol, typically over a network. The principles of service-orientation are independent of any product, vendor or technology.

SOA just makes it easier for software components over various networks to work with each other.

Web services which are built as per the SOA architecture tend to make web service more independent. The web services themselves can exchange data with each other and because of the underlying principles on which they are created, they don't need any sort of human interaction and also don't need any code modifications. It ensures that the web services on a network can interact with each other seamlessly.

**14. What is logging**

“Logging is recording every data.” Logging is keeping a record of all data input, processes, data output, and final results in a program

Log is defined as record in programming.

Example: Key logger

Key logger is an old-school hacking program which logs the user’s key presses on keyboard. When computers are installed with key logger programs, the program will record every key a user presses in the data log file or send it to a remote hacker.

**15. What is auditing**

The primary functions of an IT audit are to evaluate the systems that are in place to guard an organization's information. Specifically, information technology audits are used to evaluate the organization's ability to protect its information assets and to properly dispense information to authorized parties.

Reviewing thoroughly user time stamps, Logs, foot print, user archives, user transaction.

**16. What is a data transfer and data routing? What are some best approaches? (Split and aggregate?)**

Data transfer or transfer is any information that is transferred from one location to another through some communication method.

Data transfer refers to the secure exchange of large files between systems or organizations. In an internal context, data transfer is often used as an alternative to a holistic enterprise application integration system. However, data transfer is most often used to share data securely among business partners, suppliers, or government agencies for cooperative purposes.

A Router is a process of selecting path along which the data can be transferred from source to the destination.

**17. What is notification?**

A notification is a message, email, icon or another symbol that appears when an application wants you to pay attention

**18. What is a scheduler?**

A scheduler is a software product that allows an enterprise to schedule and track computer batch tasks. These units of work include running a security program or updating software.

Runs a program in a predefined time. Automatic trigger

**19. What is a trigger?**

Triggers are stored programs, which are automatically executed in response to certain events on a particular table or view in a database

For example, when a new record (representing a new worker) is added to the employees table, new records should also be created in the tables of the taxes, vacations and salaries. Triggers can also be used to log historical data, for example to keep track of employees' previous salaries.

Suppose you are working in a company and salary is credited in your bank account(TRIGGER) due to this trigger you receive a message “Your Salary Has Been Credited”

**20. What is a database**

**Database is a systematic collection of data. Databases support storage and manipulation of data.**

**21. What is data.**

Data is defined as facts or figures, or information that's stored in or used by a computer. An example of data is information collected for a research paper. An example of data is an email.

Data is raw, unorganized facts that need to be processed. Data can be something simple and seemingly random and useless until it is organized. When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information

**22. What is big data**

Big Data(volume, velocity, variety) is a term used to describe data that is too large and complex to store in traditional databases.

This data needs to be stored safely and securely for drawing business insights.

Hadoop, Spark, Cassandra, MongoDB are some of the important big data technologies that help in solving these issues.

**23. What is data analysis**

In simple words, data analysis is the process of collecting and organizing data in order to draw helpful conclusions from it.

The purpose of Data Analysis is to extract useful information from data and taking the decision based upon the data analysis.

Whenever we take any decision in our day-to-day life is by thinking about what happened last time or what will happen by choosing that particular decision. This is nothing but analyzing our past or future and making decisions based on it. For that, we gather memories of our past or dreams of our future. So that is nothing but data analysis. Now same thing analyst does for business purposes, is called Data Analysis.

**24. What is Datascience**

Data Science involves using automated methods to analyze massive amounts of data and to extract knowledge from them.

Data Science – Automatic process and prediction

Data science is a combination of algorithms and statistics (mathematics)

Data Science is the study of where information comes from, what it represents and how it can be turned into a valuable resource in the creation of business and IT strategies. Mining large amounts of structured and unstructured data to identify patterns can help an organization rein in costs, increase efficiencies, recognize new market opportunities and increase the organization's competitive advantage.

Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data.

Data Science is primarily used to make decisions and predictions making use of predictive causal analytics, prescriptive analytics (predictive plus decision science) and machine learning.

<https://www.edureka.co/blog/what-is-data-science/>

**25. What is statistics and why it is even necessary?**

“Statistics is a way to get information from data.”

It helps use the proper methods to collect the data, employ the correct analyses, and effectively present the results. Statistics is a crucial process behind how we make discoveries in science, make decisions based on data, and make predictions. Statistics allows you to understand a subject much more deeply.

**26. What is a server? And what’s the use ? How it is different from general laptop ?**

A server is a computer that provides data to other computers. It may serve data to systems on a local area network (LAN) or a wide area network (WAN) over the Internet.

A server is a computer equipped with specific programs and/or hardware that enables it to offer services to other computers (clients) on its network.

Many types of servers exist, including web servers, mail servers, and file servers. Each type runs software specific to the purpose of the server. For example, a Web server may run Apache HTTP Server or Microsoft IIS, which both provide access to websites over the Internet. A mail server may run a program like Exim or iMail, which provides SMTP services for sending and receiving email. A file server might use Samba or the operating system's built-in file sharing services to share files over a network.

Although any computer running the necessary software can function as a server, the most typical use of the word references the enormous, high-powered machines that push and pull data from the internet.

**27. What is software build and development?**

A “build” is when you’ve reached a point where the software you’re building is ready to be tested or released to the general public.

Software build is a process of compressing and packaging different and separate software modules or functionalities into one big object.

Compressing the entire software package

Example – gems chocolate

Software development is a process by which standalone or individual software is created using a specific programming language

**28. What is software deployment ?**

Software deployment is the process of delivering completed software to the client who ordered it or rolling out the software to consumers. Software deployment should only take place after thorough testing to ensure that all the flaws and bugs have been identified and fixed.

Deployment is nothing but installation.

**29. Cam development and deployment can be performed in same machine ( say laptop ?)**

Yes,Deployment refers to getting your program to a running state on a server. It doesn't need to be the production server. You can deploy an application/module to a testing server that is running on your own workstation or on a testing machine. You might perform many deployments during the development and testing stages of a module or application.

**30. What’s the advantage of separating out deployment phase to server?**

**31. What is table and pivot table and its uses.**Table – Rows and columns   
a Pivot Table is a tool built into Excel that allows you to summarize large quantities of data quickly and easily. Given an input table with tens, hundreds, or even thousands of rows, Pivot Tables allow you to extract answers to a series of basic questions about your data with minimal effort.

**32. Give me examples of below functions – Math, logical, data and time**Math – SUM()  
Logical : IF, AND, OR and NOT. **33. Substitute and replace function in exell**Subsititue: replace   
Replace: Position **34. Count and count blank in exell**The COUNT function is fully programmed. It counts the number of cells in a range that contain numbers and returns the result  
You can use the COUNTIF function in Excel to count cells that contain a specific value, count cells that are greater than or equal to a value, etc. **35. If else in exell but use Case When is SQL**

**36. Case when is especially used in creating new columns**

**Group all the customers by first name alphabetically   
Group all the customers by store id  
Group all the customers by active**

Postgresql

Installation Directory: C:\Program Files\PostgreSQL\10

Server Installation Directory: C:\Program Files\PostgreSQL\10

Data Directory: C:\Program Files\PostgreSQL\10\data

Database Port: 5432

Database Superuser: postgres

Operating System Account: NT AUTHORITY\NetworkService

Database Service: postgresql-10

Command Line Tools Installation Directory: C:\Program Files\PostgreSQL\10

pgAdmin4 Installation Directory: C:\Program Files\PostgreSQL\10\pgAdmin 4

Stack Builder Installation Directory: C:\Program Files\PostgreSQL\10

**CREATE TABLE public."Customers"**

**(**

**customer\_id numeric NOT NULL,**

**store\_id numeric,**

**first\_name text,**

**last\_name text,**

**email text,**

**address\_id numeric,**

**activebool boolean,**

**create\_date date,**

**last\_update text,**

**active numeric,**

**PRIMARY KEY (customer\_id)**

**)**

**WITH (**

**OIDS = FALSE**

**)**

**TABLESPACE pg\_default;**

**ALTER TABLE public."Customers"**

**OWNER to postgres;**

**IMP SQL**

**Select:** collects and displays the required results set  
**Where** clause filters the data by required conditions  
**And** – Compulsory  
**OR**- optional  
**Group by** – to identify duplicates and also to work with complex comparisons  
**DISTINCT** is one dimensional, to avoid duplicates and can only work on one column but not on \* or all columns   
Distinct function always expects only a column name and wild card characters are not accepted  
  
**AGGREGATE :** COUNT, SUM, MEDIAN , AVERAGE, MINIMUM  
For every aggregate function we should apply Groupby compulsory  
   
**The number** conditions is equal the number of ends  
**Strings or characters** should be placed in single quotes and only numbers are directly entered without quotes  
Extracts are also strings  
Strings is a collection of characters (Ex – Shiva, Srinu, Sanjay)

**IN** is used when the requirement is with a list(is a collection of some strings or characters or numbers)  
In – Always open brackets

**list**(is a collection of some strings or characters or numbers)  
List can be separated by any literal

Note: Brackets are used for either list or for aliasing. Alias names are not required for list

Query inside a Query is a sub Query – while the query is executed inner most queries are executed first and eventually executes the outer query

Simple counts are as easy as count (\*) but conditional counts or count along(other columns) are complex and so we he have to use group by

Aliasing should be kept when there is Sub Querying (reference to Question 10)

**Practice Questions and Answers**

**Databases Uses: Postgresql**

**SELECT \* FROM "Customers" – wildcard querying (do not do in production environment)  
 instead use this** **SELECT \* FROM "Customers" LIMIT 100**

1. **Load(ex :csv) and Storage**
2. **Retrieval(SQL) - Structured Query language**

* **SELECT \* FROM “Customers”**
* **SELECT \* FROM “Customers” WHERE store\_id = 2**
* **SELECT \* FROM "Customers" WHERE store\_id = 2 AND first\_name LIKE 'S%' (equal to is exact match)**
* **SELECT \* FROM "Customers" WHERE store\_id = 2 AND first\_name = 'Susan'**
* **SELECT \* FROM "Customers" ORDER BY first\_name**
* **SELECT \* FROM "Customers" WHERE first\_name LIKE 'T%' AND active = 0**
* **SELECT DISTINCT(store\_id) FROM "Customers" – to identify number of different store id’s – why distinct**
* **SELECT COUNT (\*) FROM "Customers" – Number of records in a database**
* **SELECT COUNT (\*) FROM "Customers" WHERE first\_name LIKE 'S%' – to find number**
* **SELECT COUNT (DISTINCT(first\_name)) FROM "Customers" WHERE first\_name LIKE 'S%' – to identify and remove duplicates**
* **SELECT first\_name, COUNT (\*) FROM "Customers" WHERE first\_name LIKE 'S%' GROUP BY first\_name – grouped dataset to identify duplicates with all names**
* **SELECT first\_name, COUNT (\*) FROM "Customers" WHERE first\_name LIKE 'S%' GROUP BY first\_name HAVING count (\*) > 1 - grouped dataset to identify duplicates with required names**

**1. Display all the customer information where store\_id is not 2**A) SELECT \* FROM "Customers" WHERE store\_id <> 2

**2.Display all customers where first\_name is not starting with S**

A) SELECT \* FROM "Customers" WHERE first\_name NOT LIKE 'S%'

**3.Display all customers where first\_name is not equal to susan and store\_id is not equals to 2**

1. SELECT \* FROM "Customers" WHERE first\_name <> 'SUSAN' AND store\_id <> 2

**4.Display all customer information sorted with last name**

**A)**  **SELECT \* FROM "Customers" ORDER BY last\_name**

**5. Display all inactive customers**

**A) SELECT \* FROM "Customers" WHERE active = 0**

**6.Display different active status information**

**7.How many customers are available in the customers table**

**A) SELECT COUNT (\*) FROM "Customers"**

**8.How many numbers of customers are starting with T**

**A) SELECT COUNT (\*) FROM "Customers" WHERE first\_name LIKE 'T%' OR last\_name LIKE 'T%'**

**A) SELECT first\_name, last\_name FROM "Customers" WHERE first\_name LIKE 'T%' OR last\_name LIKE 'T%'**

**9.How many customers are active in the year 2006 – get year from entire date**

**A) SELECT COUNT (\*) FROM "Customers" WHERE EXTRACT(YEAR FROM create\_date) = '2006' AND activebool = 't' AND active = 1**

**10.Display all customer information whose first name is starting with T by avoiding duplicates first name.**

**SELECT \* FROM "Customers" WHERE first\_name LIKE 'T%' AND first\_name NOT IN ?**

**Line 1 - SELECT \* FROM "Customers" WHERE first\_name NOT IN (SELECT first\_name FROM (SELECT first\_name, COUNT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%' GROUP BY first\_name HAVING COUNT(\*) > 1) a) b**

**SELECT \* FROM "Customers" WHERE last\_name IN (SELECT last\_name FROM (SELECT last\_name, COUNT(last\_name) FROM "Customers" WHERE last\_name NOT LIKE 'S%' GROUP BY store\_id HAVING COUNT(\*) > 1) a)**

**Reference queries:**

**--SELECT \* FROM (SELECT DISTINCT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%') shiva**

**--SELECT \* FROM "Customers" WHERE first\_name LIKE 'T%'**

**--SELECT first\_name FROM "Customers" WHERE first\_name LIKE 'T%'-- with duplicates--**

**--SELECT first\_name, COUNT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%' GROUP BY first\_name HAVING COUNT(\*) > 1--WITH OUT DUPLICATES--**

**--SELECT first\_name FROM (SELECT first\_name, COUNT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%' GROUP BY first\_name HAVING COUNT(\*) > 1) a**

**11.Display only the duplicated customers in the table**

**SELECT first\_name, last\_name FROM "Customers" WHERE first\_name LIKE 'T%' OR last\_name LIKE 'T%'**

**Aliasing: SELECT \* FROM (SELECT DISTINCT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%') srinu  
Sub queries should always be in () and should be given a name**

**Assuming a result as another table is a skill and that is called sub querying**

**12.**

**SELECT first\_name, COUNT(first\_name) FROM "Customers" WHERE first\_name LIKE 'T%' GROUP BY first\_name HAVING COUNT(\*) > 1**

**13. Display all customer information whose last name not starting with s by avoiding duplicates strore\_id**

SELECT \* FROM "Customers" WHERE last\_name IN (SELECT last\_name FROM (SELECT last\_name, COUNT(last\_name) FROM "Customers" WHERE last\_name NOT LIKE 'S%' GROUP BY store\_id HAVING COUNT(\*) > 1) a)

SELECT \* FROM "Customers" WHERE last\_name IN (SELECT last\_name FROM (SELECT last\_name, DISTINCT (last\_name) FROM "Customers" WHERE last\_name NOT LIKE 'S%' GROUP BY store\_id HAVING COUNT(\*) > 1) a)

**15. What is the difference between where and having**

The main difference between WHERE and HAVING clause comes when used together with GROUP BY clause, In that case WHERE is used to filter rows before grouping and HAVING is used to exclude records after grouping

On the other hand, HAVING can only be used if grouping has been performed using GROUP BY clause in the SQL query. Another worth noting thing about WHERE and HAVING clause is that WHERE clause cannot contain aggregate function like COUNT(), SUM(), MAX(), MIN(), etc but HAVING clause may contain aggregate functions.

Another worth noting the difference between WHERE and HAVING clause is that WHERE is used to impose filtering criterion on a SELECT, UPDATE, DELETE statement as well as single row function and used before group by clause but HAVING is always used after group by clause.

1. WHERE clause is used with UPDATE and DELETE clause, and HAVING clause can only be used with SELECT query.
2. WHERE clause is used for filtering rows and it applies on each and every row.
3. HAVING clause is used to filter group in SQL

Both the WHERE and the HAVING clause are very similar and are used to restrict the rows that are returned in a SELECT query.

The difference between WHERE and HAVING(after result –group by) clause are:

The WHERE clause is used to filter rows before the grouping is performed.

The HAVING clause is used to filter rows after the grouping is performed. It often includes the result of aggregate functions and is used with GROUP BY.

**16. What is CASE WHEN statement in SQL and how it is useful**

Case one only for current user and for temporarily prposes

The CASE statement goes through conditions and returns a value when the first condition is met (like an IF-THEN-ELSE statement). So, once a condition is true, it will stop reading and return the result. If no conditions are true, it returns the value in the ELSE clause.

If there is no ELSE part and no conditions are true, it returns NULL.

**17. Get all the duplicate customer details whose first name starts with S**

SELECT \* FROM "Customers" WHERE first\_name IN

(SELECT first\_name FROM

(SELECT first\_name, COUNT(first\_name) FROM "Customers" WHERE first\_name LIKE 'S%'

GROUP BY first\_name HAVING COUNT(\*) > 1)

**SELECT \* FROM "Customers" WHERE first\_name IN**

(SELECT first\_name FROM

(SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'S%'

GROUP BY first\_name HAVING COUNT(\*) > 1) a)

COUNT SHOULD ALWAYS BE AT SELECT

Limit – limits the display of number of rows  
Order by and group should always be last statement**.**

**18. Display the duplicate customers first name with t**

SELECT COUNT(\*),first\_name FROM "Customers" WHERE first\_name like 'T%' GROUP BY first\_name HAVING COUNT(\*) > 1

**19. Display all duplicated customers**

SELECT COUNT(\*),first\_name FROM "Customers" GROUP BY first\_name HAVING COUNT(\*) > 1

Sum: SELECT SUM(customer\_amount) FROM "Customers"

**20.** **Deactivate all customers starting with first name L and update in a new column (Case When)**

SELECT \*,

CASE

WHEN first\_name LIKE 'L%' THEN 0 --this condition is for active\_test coloumn only

ELSE active -- this helps to retain the original values (donot do anything)

END AS active\_test

FROM "Customers" WHERE first\_name like 'L%'; -- this condition is for whole table

**21. Deactivate all customers starting with first name L and update in a original column**

UPDATE "Customers" SET active = 0 WHERE first\_name like 'L%'

**22. DELETE all customers starting with first name Lo and whoever deactived**

DELETE FROM "Customers" WHERE first\_name like 'Lo%' AND ACTIVE = 0

**23. Drop or Truncate table**

DROP table “Customers”

**24. Display all customer data**

SELECT \* FROM "Customers"

**25. Display only first 100 records/rows of customer data**

SELECT \* FROM "Customers" limit 100

**26. Display all customer information whose first name starts with S**

SELECT \* FROM "Customers" WHERE first\_name like 'S%'

**27. Display duplicate customer information whose first name starts with S**

SELECT COUNT(\*),first\_name FROM "Customers" WHERE first\_name like 'S%' GROUP BY first\_name HAVING COUNT(\*) > 1

OR

SELECT \* FROM "Customers" WHERE first\_name IN

(SELECT first\_name FROM

(SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'S%'

GROUP BY first\_name HAVING COUNT(\*) > 1) a)

**28. Testing – Deactivate all customers whose lastname starts with T**

SELECT \*,

CASE

WHEN first\_name LIKE 'T%' THEN 0

ELSE active

END AS active\_test

FROM "Customers" WHERE first\_name like 'T%';

**29. Real – Deactivate all customers whose lastname starts with T**

UPDATE "Customers" SET active = 0 WHERE first\_name like 'T%'

**30. Testing – Deactivate all customers whose last name starts with T and also duplicated with same first name**

SELECT \*,

CASE

WHEN first\_name LIKE 'T%' THEN 0

ELSE active

END AS active\_test

FROM "Customers" WHERE first\_name IN

(SELECT first\_name FROM

(SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'T%'

GROUP BY first\_name HAVING COUNT(\*) > 1) a)

**31. Real - Deactivate all customers whose lastname starts with T and also duplicated with same first name**

**--Deactivate all customers whose first name starts with Te and also duplicated with same first name starts with T**

**-- update + \* + first name starts with Te + duplicate with first name starts with T**

UPDATE "Customers" SET active = 0

--SELECT \* FROM "Customers"

WHERE first\_name like 'Te%'

AND first\_name IN

(SELECT first\_name FROM (SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'T%'

GROUP BY first\_name HAVING COUNT(\*) > 1) a)

**SELECT address\_id FROM "Customers"**

**WHERE first\_name like 'Te%'**

**AND first\_name IN**

**(SELECT first\_name FROM (SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'T%'**

**GROUP BY first\_name HAVING COUNT(\*) > 1) a)**

**31. Update QLD with specials using case when**

**SELECT \*,**

**CASE WHEN "district" = 'QLD' THEN 'SPECIAL'**

**ELSE 'NONSPECIAL'**

**END AS "SP"**

**FROM "Address"**

**31. Update store ID**

UPDATE "Customers" SET "store\_id" = 3 WHERE "store\_id" = 2

**32. Why do you need joins?**

To get more detailed information from corresponding tables

List is always one dimensional and consists of only one column   
Ex: bullet points or coma separated

**33. Get all duplicated customer address details – Method 1**

**all customers whose first name starts with Te and also duplicated with same first name starts with T**

**\*(address)+ first name starts with Te(customer table) + duplicate first name starts with T(customer table) + customer address details(address table)**

Method 1: SELECT \* FROM "Address" WHERE address\_id IN

(SELECT address\_id FROM "Customers"

WHERE first\_name like 'Te%'

AND first\_name IN

(SELECT first\_name FROM (SELECT COUNT(\*), first\_name FROM "Customers" WHERE first\_name LIKE 'T%'

GROUP BY first\_name HAVING COUNT(\*) > 1) a)

Method2:

**Possibilities**

1. **Joining table can have exact number of rows(inner join) – Gets only common key values**

**SELECT \***

**FROM "Customers"**

**INNER JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id;**

**SELECT DISTINCT(A.address\_id)**

**FROM "Customers" A**

**INNER JOIN "Address" B**

**ON A.address\_id = B.address\_id;**

1. **Joining table can have less number of rows(left join) - gets left table values along with common keys**

**SELECT \***

**FROM "Customers"**

**LEFT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id;**

1. **Joining table can have more number of rows (right join) – gets right table values along with common keys**

**SELECT \***

**FROM "Customers"**

**RIGHT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id;**

1. **Joining table with all the number of rows from both the tables(outer join)**
2. **Joining table can have zero number of rows**

**SELECT COUNT(\*), A.address\_id**

**FROM "Customers" A**

**INNER JOIN "Address" B**

**ON A.address\_id = B.address\_id GROUP BY A.address\_id HAVING COUNT(\*) > 1**

**599**

**595**

**600**

**596**

**593**

**602**

**604**

**597**

**605**

**594**

**601**

**592**

**603**

**591**

**598**

**5**

**SELECT \***

**FROM "Customers" A**

**INNER JOIN "Address" B**

**ON A.address\_id = B.address\_id WHERE A.address\_id IN**

**(SELECT address\_id FROM (SELECT COUNT(\*), A.address\_id**

**FROM "Customers" A**

**INNER JOIN "Address" B**

**ON A.address\_id = B.address\_id GROUP BY A.address\_id HAVING COUNT(\*) > 1) srinu)**

**Primary Key :**

**INSERT INTO "Customers" VALUES (1001, 1, 'Mary','Smith', 'mary.smith@sakilacustomer.org', 5, true, '2006-02-14', '49:45.7', 1 )**

**34. when you wanted to get entire row of duplicate**

SELECT COUNT (\*), first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update, active FROM "Customers" GROUP BY store\_id, first\_name, last\_name, email, address\_id, activebool, create\_date, last\_update, active HAVING COUNT (\*) > 1

**35. Number of distinct records**

SELECT COUNT (DISTINCT (address\_id)) from "Address"

--SELECT \*   
--FROM "Customers"  
--INNER JOIN "Address"  
--ON "Customers".address\_id = "Address".address\_id;  
--SELECT COUNT(DISTINCT(address\_id)) FROM "Customers" – 592  
--SELECT COUNT(DISTINCT(address\_id)) FROM "Address" -- 603

**36. SHOW ME ALL non matching CUSTOMRE ADDRESS ID BETWEEN CUSTOMERS AND ADDRESS**

--\* + NOT IN CUSTOMER ADDRESS ID( customer, address)  
--SELECT \* FROM "Address" WHERE address\_ID NOT IN ( SELECT address\_id FROM "Customers")--603(adress) - 592(Customer)  
--Show me all matching customer address ID between customer and address  
SELECT \* FROM "Address" WHERE address\_ID IN ( SELECT address\_id FROM "Customers")

**Inner Join**

1. **address\_id is the common column in both the tables – Inner Join**
2. **Total count of address table is 604 and customer table is 609**
3. **Total distinct of address table 603 and customer table is 592 – Imp(remove the duplicate address\_id)**
4. **Principle: Total distinct of Inner join of both the tables is 592 – brings only common rows b/w both the tables**
5. **We can display the inner join results of 592 by only avoiding duplicates**

**SELECT COUNT(\*), "Customers".address\_id**

**FROM "Customers"**

**INNER JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id**

**GROUP BY "Customers".address\_id**

1. **Get the first name, last name and phone numbers of duplicated address\_id customers**

**--**Get the first name, last name and phone numbers of duplicated address\_id customers

--display first name, last name and phone numbers + duplicate address\_id(GROUP BY)

SELECT first\_name, last\_name, phone, "Customers".address\_id FROM "Customers"

INNER JOIN "Address"

ON "Customers".address\_id = "Address".address\_id

Where "Customers".address\_id IN

(Select address\_id FROM (SELECT COUNT(\*), "Customers".address\_id

FROM "Customers"

INNER JOIN "Address"

ON "Customers".address\_id = "Address".address\_id

GROUP BY "Customers".address\_id

HAVING COUNT(\*) > 1) sanjay)

1. **To combine the out puts of two different queries we have to either use IN or Where**

**Left Join((Or Left-outer join)**

1. **address\_id is the common column in both the tables – Left Join**
2. **Total count of customer table is 609(Left table) and address table is 604 (Right Table)**
3. **Total distinct of customer table is 592 and address table 603 – Imp(remove the duplicate address\_id)**
4. **Principle: Total Left join of both the tables is 609 – brings common rows and also left table row((including left table NULL values)**
5. **Principle: Total distinct of Left join of both the tables is 609 or more possibility? – brings common rows and also left table rows**
6. **We can display the left join results of 609 or more possibility?**

**SELECT COUNT(\*)**

**FROM "Customers"**

**LEFT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id**

1. **Get the first name starts with P, last name, email and activebool not equals to zero where storeid is 2 and duplicated address\_id customers**

**--1**

**--\* first name starts with P**

**--\* last name**

**--\* email**

**--\* activebool not euat to 0**

**--\* storeid = 2**

**SELECT first\_name,last\_name,email,activebool,store\_id**

**FROM "Customers" WHERE first\_name like 'P%' AND activebool !=false AND store\_id = 2 AND**

**address\_id IN**

**(--duplicated address\_id customers(Group by)**

**SELECT address\_id FROM**

**(SELECT COUNT(\*), "Customers".address\_id**

**FROM "Customers"**

**LEFT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id**

**GROUP BY "Customers".address\_id**

**HAVING COUNT(\*) > 1) Tooty)**

**7. To combine the out puts of two different queries we have to either use IN or Where**

**RIGHT Join(or Right-Outer Join)**

1. **address\_id is the common column in both the tables – Right Join**
2. **Total count of customer table is 609(Left table) and address table is 604(right table)**
3. **Total distinct of customer table is 592 and address table 603 – Imp(remove the duplicate address\_id)**
4. **Principle: Total Right join of both the tables is 620 – brings common rows and also left table row(including right table NULL values)**
5. **Principle: Total distinct of Right join of both the tables is 609– brings common rows and also left table row**
6. **We can display the Right join results of 609**

**SELECT DISTINCT(COUNT("Customers".address\_id))**

**FROM "Customers"**

**RIGHT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id**

1. **Get district starting with S, address, city\_id and duplicated address\_id**

**SELECT district, address, city\_id**

**FROM "Address" WHERE district like 'S%' AND**

**address\_id IN**

**(--duplicated address\_id customers(Group by)**

**SELECT address\_id FROM**

**(SELECT COUNT(\*), "Customers".address\_id**

**FROM "Customers"**

**RIGHT JOIN "Address"**

**ON "Customers".address\_id = "Address".address\_id**

**GROUP BY "Customers".address\_id**

**HAVING COUNT(\*) > 1) Rainy)**

**Types of sub query intersections or joining different answers**

1. **IN (List for second query only)**
2. **=, >, < (Single Value for the second query only)**
3. **<> or !=, >=, <= (Single value for the second query only)**

**Clarifying Questions/concerns/doubts on the final result are called TEST CASES**

**Ex: -- Question**

**--SELECT**

**--\* first name starts with P**

**--\* last name**

**--\* email**

**--\* activebool not euat to 0**

**--\* storeid = 2**

**Answer**

**Test case 1 - Verify whether activebool = false**

**--SELECT \* FROM**

**--(SELECT first\_name, last\_name, email, activebool, store\_id FROM "Customers"**

**-- WHERE first\_name like 'P%' AND activebool != false AND store\_id = 2) Shiva WHERE activebool = false**

**--Test case 2 -Verify whether or not first name starting with P**

**SELECT \* FROM**

**(SELECT first\_name, last\_name, email, activebool, store\_id FROM "Customers"**

**WHERE first\_name like 'P%' AND activebool != false AND store\_id = 2) Shiva WHERE first\_name NOT like 'P%'**

**Full Outer Join(Or Left - Outer Join + Right – Outer Join) (Or Left – Outer Join UNION Left – Outer Join)**

**select \***

**from "Customers" A**

**FULL OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**-------------------------- OR -----------------------------------------**

**select \***

**from "Customers" A**

**LEFT OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**UNION**

**select \***

**from "Customers" A**

**RIGHT OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**Full Outer Join Avoiding NULLs(Or Left - Outer Join + Right – Outer Join) (Or Left – Outer Join UNION Left – Outer Join)**

**select \***

**from "Customers" A**

**FULL OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**WHERE**

**A.address\_id IS NOT NULL AND B.address\_id IS NOT NULL**

**------------------------------------------------ OR --------------------------------------------**

**select \***

**from "Customers" A**

**LEFT OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**UNION**

**select \***

**from "Customers" A**

**RIGHT OUTER JOIN**

**"Address" B**

**ON**

**A.address\_id = B.address\_id**

**WHERE A.address\_id IS NOT NULL AND B.address\_id IS NOT NULL**

**CRUD Operations – Create/read(select)/update/delete**

**ACID Properties – Atomicity/Consistency/Isolation/Durability**

Atomicity: The atomicity acid property in SQL. It means either all the operations (insert, update, delete) inside a transaction take place or none. Or you can say, all the statements (insert, update, delete) inside a transaction are either completed or rolled back.

Consistency guarantees that a transaction never leaves your database in a half-finished state.  
Isolation keeps transactions separated from each other until they’re finished.  
Durability guarantees that the database will keep track of pending changes in such a way that the server can recover from an abnormal termination.

**Motivation**

**Database Query Language(SQL)** - to organize, compute(Case When, Group by, Having), store(import and load to data base), retrieve(Select) and display the data with in the sql server or local machine.**SQL/TSQL (simple, intermediate)**PLSQL (advanced)Data Mining, Statistics (super-advanced)

**Programming language(Python/Java etc)** - to organize, compute, store, retrieve and display the data across the globe or with in the organization (from DB) utilizing http internet protocol.**HTML, CSS**, JSP —> Display

Servlet, JavaScript, JSTL, AngularJS, Node, ExpressJS —> organize, compute, store and **retrieve (simple to intermediate)**

Cobol, Dos(windows), Bash scripting(Linux), Java, .Net, **Python,** Scala, Ruby on Rails—> organize, compute, store and retrieve **(advanced)**

SaS, IBM data stage, IIB, Bigdata ETL—> organize, compute, store and retrieve **(super-advanced)**

AI, RPA, Machine Learning and Data Science —> organize, compute, store and retrieve **(super-advanced but still experimental)**

Gaming, Compilers and Virtualization —> organize, compute, store and retrieve **(super-advanced with proven experiments)** - Data manipulation will be on bits, bytes, frames and pixels.

VLSI, embedded systems and circuit designing —> organize, compute, store and retrieve **—> (super-advanced with proven experiments but on hardware)**

**--SELECT \* FROM "Address"**

**--UPDATE "Address" SET "district" = 'Alberta' WHERE "district" = 'Gilberta'**

**SELECT \*,**

**CASE**

**WHEN "city\_id" > 0 and "city\_id" <=100 THEN 'ALPHA'**

**WHEN "city\_id" > 100 and "city\_id" <=300 THEN 'BETA'**

**WHEN "city\_id" > 400 and "city\_id" <=500 THEN 'DELTA'**

**ELSE 'GAMA'**

**END AS "state"**

**FROM "Address"**

**Case when**

**--Count aggregate function**

**--SELECT COUNT(first\_name),first\_name FROM "Customers" GROUP BY first\_name HAVING COUNT(first\_name)> 1**

**--SELECT COUNT(store\_id), store\_id FROM "Customers" GROUP BY store\_id HAVING COUNT(store\_id)> 1**

**SELECT address\_id, order\_amount FROM "Customers"**

**--UPDATE "Customers" SET order\_amount= 2000 WHERE address\_id> 200**

**--SELECT \*,**

**--CASE**

**--WHEN address\_id>200 THEN 2000**

**--ELSE 100**

**--END AS "modified\_orderamt"**

**--FROM "Customers"**