PLACEMENT REFRESHER PROGRAM

Session 6 - SQL 4

Advanced Database Concepts & Problem Solving

By
Ritesh Kumar Pandey

Agenda

- Normalization
- Indexing
- Views
- Interview Questions & Problem Solving

Normalization

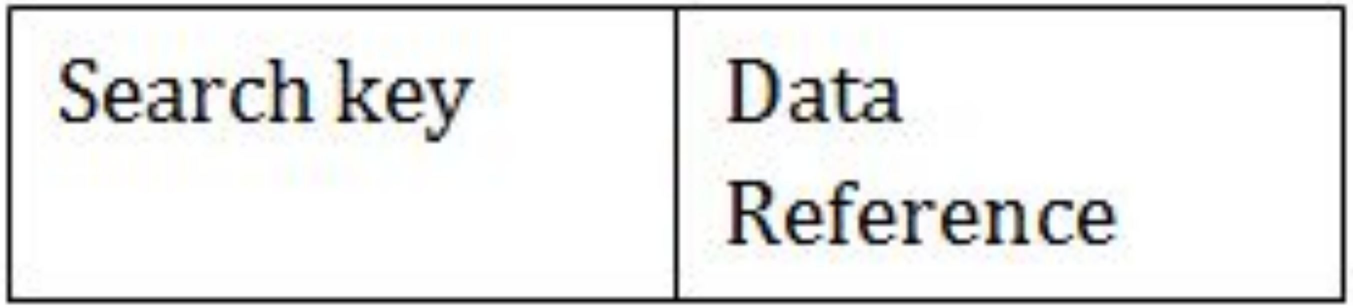
- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.
- Normalization divides the larger table into smaller and links them using relationships.
- The normal form is used to reduce redundancy from the database table.

Normalization - Types

Normal Form	Description
1NF	A relation is in 1NF if it contains an atomic value.
2NF	A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.
3NF	A relation will be in 3NF if it is in 2NF and no transition dependency exists.
BCNF	A stronger definition of 3NF is known as Boyce Codd's normal form.

Indexing

- Indexing is used to optimize the performance of a database by minimizing the number of disk accesses required when a query is processed.
- The index is a type of data structure. It is used to locate and access the data in a database table quickly.

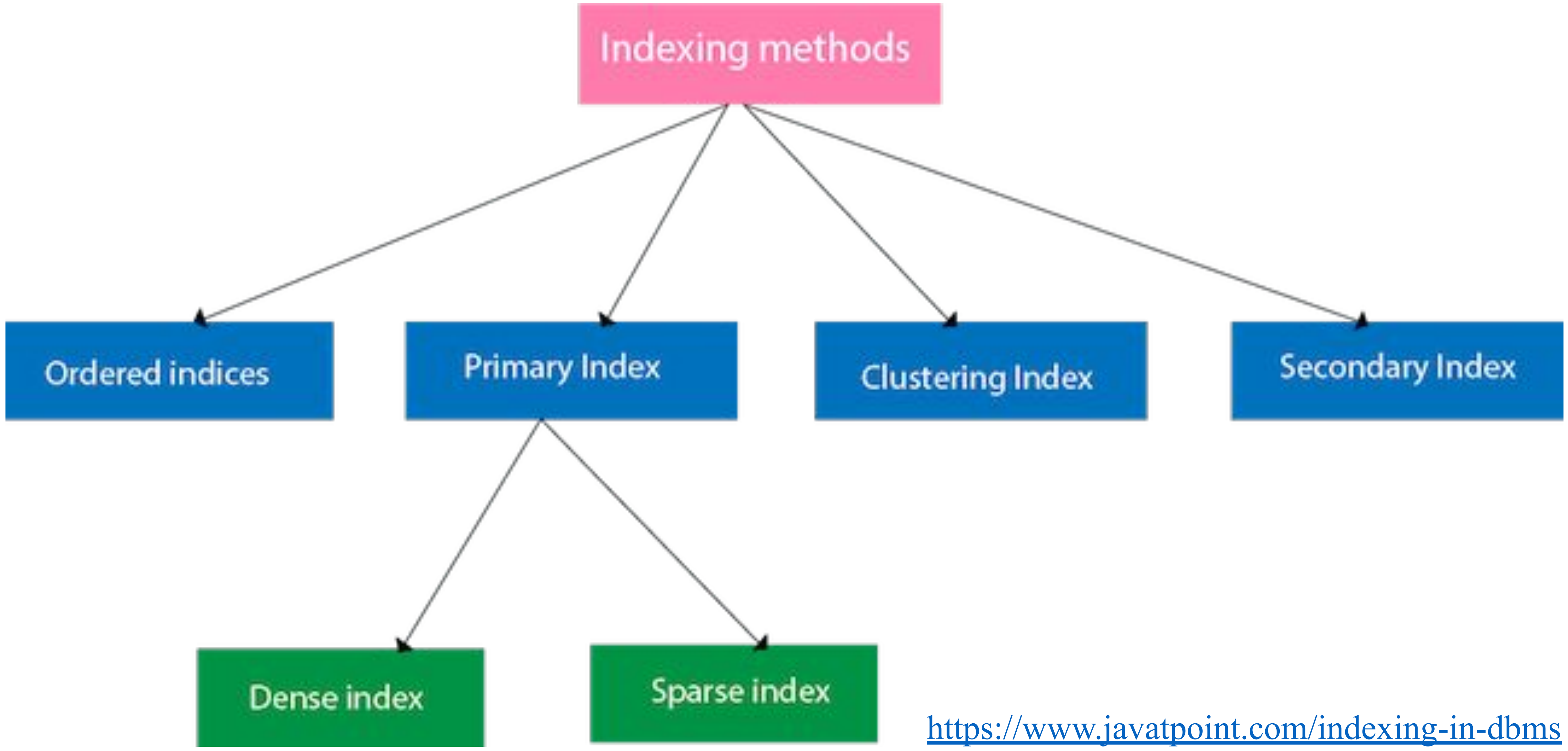


Search key	Data Reference

Fig: Structure of Index

Indexing - Types

upGrad



https://www.javatpoint.com/indexing-in-dbms

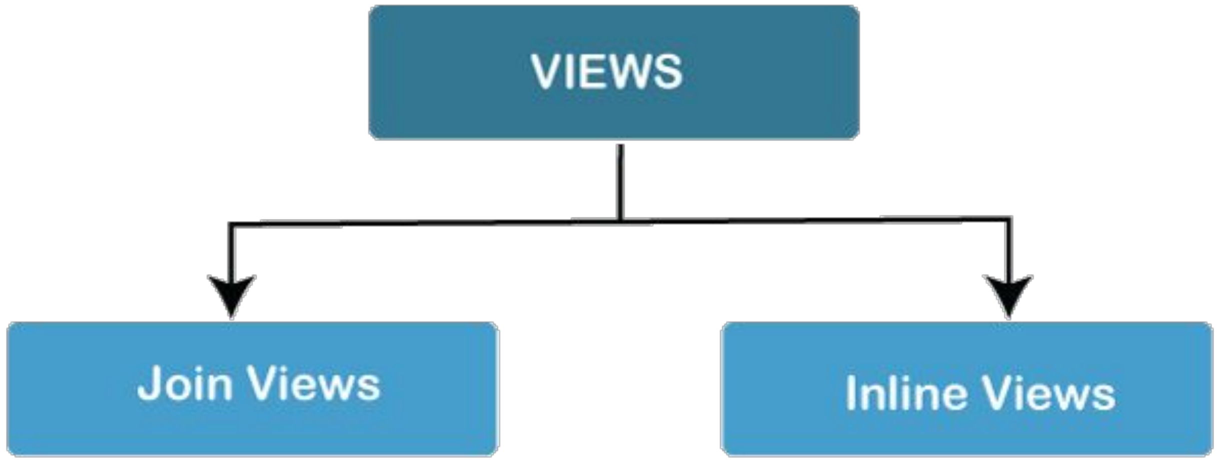
Views

- Views in SQL are considered as a virtual table. A view also contains rows and columns.
- To create the view, we can select the fields from one or more tables present in the database.
- A view can either have specific rows based on certain condition or all the rows of a table.

Significance of Views:
Views are highly significant, as they can provide advantages over tasks. Views can represent a subset of data contained in a table. Consequently they can limit the degree of exposure of the underlying base table to the outer world. They are used for security purpose in database and act as an intermediate between real table schemas and programmability. They act as aggregate tables.

Views - Types

upGrad



- **Join View:** A join view is a view that has more than one table or view in its from clause and it does not use any Group by Clause, Rownum, Distinct and set operation.

- **Inline View:** An inline view is a view which is created by replacing a subquery in the from clause which defines the data source that can be referenced in the main query. The sub query must be given an alias for efficient working.

Interview Questions

upGrad

1. What is SQL, and why is it important in the database world?
2. Explain the difference between SQL and NoSQL databases.
3. Define ACID properties in the context of database transactions.
4. What is normalization, and why is it essential in database design?
5. Explain the difference between a primary key and a foreign key.
6. What is the purpose of an index in a database table?
7. Describe the difference between a one-to-one and a one-to-many relationship in database design.
8. What is the GROUP BY clause, and why is it used?
9. How can you combine the results of two or more SELECT statements into one result set?
10. What is the purpose of the ORDER BY clause, and how do you use it to sort query results?

Interview Questions

upGrad

1. Define CRUD operations in SQL, and list the actions they represent.
2. What is the purpose of the SQL DELETE statement, and when is it commonly used?
3. What is an SQL JOIN, and how does it work?
4. Explain the different types of JOIN operations in SQL.
5. How do you create a self-JOIN, and when is it useful?
6. What is the difference between JOIN and UNION in SQL, and when would you use each?
7. Write an SQL subquery to find the second-highest salary in an "employees" table.
8. Explain the concept of correlated subqueries and provide an example.
9. What are EXISTS and IN operators in SQL subqueries, and how do they differ?
10. Explain the purpose of the HAVING clause in SQL and how it differs from the WHERE clause.

Interview Questions

1. Differentiate between the SQL COUNT(*) and COUNT(column_name) functions and provide examples.
2. What are window functions, and how are they used in SQL queries?
3. Describe the usage of common table expressions (CTEs) in SQL.
4. What are SQL indexes, and why are they important for query performance?
5. What is the Cartesian product in SQL, and why should you avoid it?
6. Explain how to handle null values in SQL queries and why it's important.
7. Describe a situation where a LEFT JOIN might return unexpected results.
8. Provide examples of how SQL is used in real-world scenarios, such as e-commerce, finance, or healthcare.
9. How can SQL be utilized for data analysis and reporting in business intelligence?
10. In what ways can SQL be applied in data migration and ETL (Extract, Transform, Load) processes?

Interview Questions - Query Based

upGrad

Find the total population of the "North America" continent.

Write an SQL query to find the official language of each country with a population greater than 100 million.

Interview Questions - Query Based

upGrad

Find the total population of the "North America" continent.

```
SELECT SUM(population)
FROM Country
WHERE continent = 'North America';
```

Write an SQL query to find the official language of each country with a population greater than 100 million.

```
SELECT c.name, cl.language
FROM Country c
JOIN countrylanguage cl ON c.code = cl.countrycode
WHERE cl.isofficial = 'T' AND c.population > 100000000;
```

Interview Questions - Query Based

Find the countries with the highest and lowest populations in each continent.

Interview Questions - Query Based

upGrad

Find the countries with the highest and lowest populations in each continent.

```
SELECT c1.continent, c1.name AS highest_population_country, c2.name AS lowest_population_country
FROM (
    SELECT continent, name, population
    FROM Country
    WHERE population = (SELECT MAX(population) FROM Country c2 WHERE c2.continent =
Country.continent)
) c1
JOIN (
    SELECT continent, name, population
    FROM Country
    WHERE population = (SELECT MIN(population) FROM Country c2 WHERE c2.continent =
Country.continent)
) c2
ON c1.continent = c2.continent;
```

Interview Questions - Query Based

List the cities in each country where more than 20% of the population speaks the official language.

Interview Questions - Query Based

upGrad

List the cities in each country where more than 20% of the population speaks the official language.

```
SELECT c.name AS country, ct.name AS city
FROM City ct
JOIN Country c ON ct.countrycode = c.code
WHERE ct.population * 0.2 >= ANY (
    SELECT cl.percentage
    FROM CountryLanguage cl
    WHERE cl.countrycode = c.code AND cl.isofficial = 'T'
);
```

Interview Questions - Query Based

Write an SQL query to find the average population of all cities in each country.

Find the total population of the world.

Interview Questions - Query Based

upGrad

Write an SQL query to find the average population of all cities in each country.

```
SELECT c.name AS country, AVG(ct.population) AS average_population
FROM City ct
JOIN Country c ON ct.countrycode = c.code
GROUP BY c.name;
```

Find the total population of the world.

```
SELECT SUM(population)
FROM Country;
```

Interview Questions - Query Based

List the countries where English is an official language and spoken by more than 10% of the population.

Interview Questions - Query Based

List the countries where English is an official language and spoken by more than 10% of the population.

```
SELECT c.name
FROM Country c
WHERE 'English' IN (
    SELECT language
    FROM CountryLanguage
    WHERE countrycode = c.code AND isofficial = 'T'
) AND 'English' IN (
    SELECT language
    FROM CountryLanguage
    WHERE countrycode = c.code AND percentage > 10
);
```

Interview Questions - Query Based

Write an SQL query to find the continents where no country has a population greater than 500 million.

Interview Questions - Query Based

upGrad

Write an SQL query to find the continents where no country has a population greater than 500 million.

```
SELECT DISTINCT c1.continent
FROM Country c1
WHERE NOT EXISTS (
   SELECT 1
   FROM Country c2
   WHERE c2.continent = c1.continent AND c2.population > 500000000
);
```

THANK YOU