|  |  |
| --- | --- |
| **## SQL Study Guide** | |
| **### SQL** |  |
| - What is SQL | Structured Query Language |
| - What are the SQL sublanguages; | **DDL – Data Definition Language**  CREATE, ALTER, DROP and TRUNCATE  **DML – Data Manipulation Language**  INSERT, (SELECT\*), UPDATE and DELETE  **DQL\* - Data Query Language**  SELECT  **DCL – Data Control Language**  GRANT and REVOKE  **TCL – Transaction Control Language**  COMMIT, ROLLBACK, SAVEPOINT, RELEASE SET  TRANSACTION. |
| - What are the Key Statements for the Sub Languages |  |
| - What is multiplicity | **Multiplicity** establishes the bounds for data. |
| - What is cardinality | **Cardinality** describes the actual relationship for data. |
| - What is a Primary Key | A **primary key** is a column of table which uniquely identifies each tuple (row) in that table. Primary key enforces integrity constraints to the table. Only one primary key is allowed to use in a table. The primary key does not accept the any duplicate and NULL values. The primary key value in a table changes very rarely so it is chosen with care where the changes can occur in a seldom manner. A primary key of one table can be referenced by foreign key of another table. |
| - What is a Foreign Key | A **foreign key** is a column or group of columns in a relational database table that provides a link between data in two tables. It is a column (or columns) that references a column (most often the primary key) of another table. |
| - What is referential integrity | is a relational database concept, which states that table relationships must always be consistent. |
| - What are the different constraints | Integrity constraints are mainly enforced to maintain the data consistency in database as they restrict the data that can go into the table. These integrity constraints are categorized into two categories **column level** and **table level** constraints. Mainly used constraints are:  **not null constraint**: Ensures that a column cannot have NULL value.  **Default**: Provides a default value for a column when none is specified.  **Unique**: Ensures that all values in a column are different.  **Primary Key**: Uniquely identified each rows/records in a database table.  **Foreign Key**: Uniquely identified a rows/records in any another database table(fetch the value from the column of another table that is the primary key in that table).  **Check constraint**: The CHECK constraint ensures that all values in a column satisfy certain conditions that are mentioned in the check condition. |
| - What do you call a record with a foreign key relationship that doesn’t exist |  |
| - What are the differences between WHERE vs HAVING | The **WHERE clause** works on row’s data, not on aggregated data.    The **HAVING** clause works on aggregated data. |
| - what are the differences between GROUP BY and ORDER BY | **GROUP BY statement** is used to group the rows that have the same value. It is often used with aggregate functions.  **ORDER BY keyword** sort the result-set either in ascending or in descending order. |
| - What is the difference between an aggregate function and a scalar function? | **Aggregate function** - Functions that are used to do operations from the values of the column and a single value is returned.  **Scalar Function** - Functions that are based on user input that return a single value each time it’s invoked. |
| - Name some scalar functions and their usage | UCASE() - It converts the value of a field to uppercase  LCASE() - It converts the value of a field to uppercase  MID() - extracts texts from the text field.  LEN() - returns the length of the value in a text field.  ROUND() - used to round a numeric field to the number of decimals specified.  NOW() - the current system date and time  FORMAT() - used to format how a field is to be displayed. |
| - Name some aggregate functions and their usage | AVG() - returns average value after calculating from values in a numeric column  COUNT() - used to count the number of rows returned in a SELECT statement.  FIRST() - returns the first value of the selected column  LAST() - returns the last value of the selected column.  MAX() - returns the maximum value of the selected column.  MIN() - returns the minimum value of the selected column  SUM() - returns the sum of all the values of the selected column. |
| - What does LIKE do? | It is useful when you want to search rows to match a specific pattern, or when you do not know the entire value. |
| - How do I use sub queries? |  |
| - How does BETWEEN work? | The **BETWEEN** and **AND** operators are used in WHERE clauses to compare data for a range of values. |
| - What is the order of operations in an SQL statement? | 1. FROM 2. WHERE 3. GROUP BY 4. HAVING 5. SELECT 6. ORDER BY |
| - What are the different joins in SQL? | **INNER JOIN -**  **LEFT JOIN -**  **RIGHT JOIN -**  **FULL JOIN -** |
| - What are the different set operations in SQL? | **Set Union -**  **Set Intersection -**  **Set Difference -** |
| - Difference between union and union all |  |
| - What is the difference between joins and set operations? |  |
| - How can I create a alias in SQL | as alias\_name |
| - What does the AS keyword do in a query? | Aliases are the temporary names given to table or column for the purpose of a particular SQL query. It is used when name of column or table is used other than their original names, but the modified name is only temporary. |
| - What are the properties of a transaction? | **Atomicity Consistency Isolation Durability**  **Atomicity** - The entire transaction takes place at once or doesn’t happen at all. The transaction will be aborted and the change reverted.  **Consistency** - The database must be consistent before and after the transaction. No transaction should have any adverse effect on the data residing in the database/table.  **Isolation -** Multiple transactions occur independently without interference.  **Durability** - The changes of a successful transaction occurs even if the system failure occurs. |
| - What are the transaction isolation levels and what do they prevent? | **Read Uncommitted** - Read Uncommitted is the lowest isolation level. In this level, one transaction may read not yet committed changes made by other transaction, thereby allowing dirty reads. In this level, transactions are not isolated from each other.  **Read Committed** - This isolation level guarantees that any data read is committed at the moment it is read. Thus it does not allows dirty read. The transaction holds a read or write lock on the current row, and thus prevent other transactions from reading, updating or deleting it.  **Repeatable Read** - This is the most restrictive isolation level. The transaction holds read locks on all rows it references and writes locks on all rows it inserts, updates, or deletes. Since other transaction cannot read, update or delete these rows, consequently it avoids non-repeatable read.  **Serializable** - Highest Isolation Level. An execution of operations in which concurrently executing transactions appears to be serially or sequentially executing. |
| - What are dirty reads, non repeatable reads, and phantom reads? | **Dirty Read** - when a transaction reads data that has been added by a different transaction that has not yet been committed.  **Non Repeatable Read** - occurs when a transaction reads the same row twice, and get a different value each time. Transactions re-read data that it has previously read, and finds another committed transaction as modified or deleted data.  **Phantom Read** - occurs when two same queries are executed, but the rows retrieved by the two, are different. Transaction returns a query to find that the number of records that satisfies a certain condition has changed. |
| - What is normalization | Normalization is the method used to eliminate or reduce redundancy in database tables. |
| - What are the requirements for the different normalization levels | **0NF** - Information is just put into a DB with no meaning or planning. No normalization.  **1NF** - Every attribute in that relation is singled valued attribute. All data must be atomic (smallest possible data possible while maintaining all the information). Each record should be unique. All tables should have a primary key  **2NF** - relation must be in first normal form and relation must not contain any partial dependency (If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.)  **3NF** - if there is no transitive dependency for non-prime attributes as well as it is in second normal form. No columns are dependent on a column that is not the primary key. |
| - What is pl/pgsql | Procedural Language/SQL - PL/SQL is a block structured language that enables developers to combine the power of SQL with procedural statements.All the statements of a block are passed to oracle engine all at once which increases processing speed and decreases the traffic.  **Disadvantages of SQ**L:  SQL doesn’t provide the programmers with a technique of condition checking, looping and branching.  SQL statements are passed to Oracle engine one at a time which increases traffic and decreases speed.  SQL has no facility of error checking during manipulation of data.  **Features of PL/SQL**:  PL/SQL is basically a procedural language, which provides the functionality of decision making, iteration and many more features of procedural programming languages.  PL/SQL can execute a number of queries in one block using single command.  One can create a PL/SQL unit such as procedures, functions, packages, triggers, and types, which are stored in the database for reuse by applications.  PL/SQL provides a feature to handle the exception which occurs in PL/SQL block known as exception handling block.  Applications written in PL/SQL are portable to computer hardware or operating system where Oracle is operational.  PL/SQL Offers extensive error checking. |
| - What are triggers | Are stored procedure in database which automatically invokes whenever a special event in the database occurs. |
|  |  |
| **### MAVEN** | |
| - What is Maven? |  |
| - What are the Maven Lifecycles? |  |
| - What is the purpose of the POM.xml? |  |
|  |  |
| **### Design Patterns** | |
| - What is the difference between Singleton and Factory? |  |
|  |  |
| **### JDBC** |  |
| - What is JDBC |  |
| - What are the interfaces and classes of JDBC |  |
| - What are the different types of statements |  |
| - What is SQL Injection |  |
| - What is a DAO |  |
|  |  |
|  |  |