Q.1 What are ANSI SQL Isolation levels and explain which isolation level best fits your project.

ANSI SQL isolation level can be defined as a level of isolation of transactions. Meaning a given transaction should be isolated from data modification that is created by other concurrent transactions.

There are 4 levels in isolation level.

1. Read Uncommitted
2. Read committed
3. Repeatable Read
4. Serializable

Three events are either allowed or not allowed in a given SQL level they are:-

Dirty Read in this event user is permitted to read data that is not committed in this event the data integrity is compromised and the foreign keys are also violated.

Non repeatable read: in this read the data which is to be read twice in the same transaction may or may not have the same values when it is being read for the second time.

Phantom read: this is similar to a non-repeatable read but in this, the data read the first time is not changed but more rows can be added when read for the second time.

Read uncommitted.

In reading uncommitted user is allowed for the dirty read. The transaction doesn’t have shared lock thus allowing other transaction running at the same time to modify the data which is yet to be committed by the user.

For example, a purchase is made in a table at time t1 thus increasing count, but the purchase is not yet committed. then count transaction is run by the user but during the running of the transaction, another purchase is done so we will have a different result in the count function.

Read Committed

In this Isolation level, an exclusive lock is issued by the transaction at the time when data is being modified. So transactions such as count can’t read the data which is not committed. Hence preventing the issue from the dirty read.

Example. This can prevent data integrity of sale figures of companies like Amazon.

Repeatable read

Repeatable read is similar to read committed and here also a user cannot read the data which is not committed by the user. The main difference is until the user has committed the data the repeatable read prevents the modification of the data by other users.

The use of this is when suppose a student has applied for multiple re-evaluations of his exam or group of students have applied revaluation until unless the one-course subject is not updated other revaluation will not be executed.

Serializable read

Serializable read is the most restrictive level in comparison to all previous isolation it also prevents dirty read and not repeatable read. the difference here from the repeatable read is that the user cannot modify data and it prevents phantom reads as well. A wide range of locks is placed in the data.

For example for very secure transactions such as one for the govt about crucial space research data, serializable read is used to provide maximum data integrity.

In our database Read Committed to our database will prevent dirty read so that if one transaction is happening at one time another transaction done by someone cant modify the data. Until it is committed, thus preserving data integrity.

Q2.

Lead and Lag

The lead and lag are two functions in SQL which are used to fetch data from rows. The Lead function fetches data from the next row whereas the lag function fetches data from the previous row. In other words to fetch data from subsequent row we use lead, and to fetch data from the previous row we use lag.

Many arguments can be passed in lead and lag functions such as

partition by supposing we have data from several years, but we currently only require data from the certain year we can use partition by to separate year data into decades so that we can better analyze the data.

Offset creates a physical offset in the table from the current row if the parameter is omitted. The default value of this argument is one.

Default

When the offset goes outbound from the table then we get default in return, if it is not set then the default value is null.

Order by

this is a simple argument to sort data shown by lag or lead in ascending or descending order. By default ascending order is used.

|  |  |  |
| --- | --- | --- |
| Student under professor | Count student | Previous semester count |
| Fall | 200 | 0 |
| spring | 300 | 200 |
| summer | 250 | 300 |
| fall | 400 | 250 |

|  |  |  |
| --- | --- | --- |
| Student under professor | Count student | next semester count |
| Fall | 200 | 300 |
| spring | 300 | 250 |
| summer | 250 | 400 |
| fall | 400 | NULL |

Given that in our project suppose we have to see no. of a student under a given professor in a particular semester we can use the lead and lag function to represent student under him for next semester as well as for the previous semester. This would lead to better access and representation of our data. So that professor can analyze if students are increased or decreased under him or what kind of coursework he can give based on team formation for projects.

3. What is a deadlock and how it can be avoided. Based on the data model of your project do you think deadlock will arise? Explain the justification in your own words

A deadlock can occur when both a and b are performing a transaction at the same time and have implemented a lock on both resources thus creating a deadlock. For Example, suppose X is trying to make a transaction from his account X1 to Y1 at the same time Y is trying to make a transaction from his account Y1. Both have resources X2 and Y2 used for a transaction now x will try to lock X2 and Y2 whereas Y1 will apply a lock on X2 and Y2. Now, the X1 process won't release lock as it will wait for Y1 to complete, and Y1 won't release lock as it will wait for X1 to complete. Thus creating a deadlock

There are several ways to avoid deadlock some of which are

1. Not allowing input by the user during the transaction
2. Making sure the design of the database is normalized.
3. Reducing the length of transactions
4. Using bound connections
5. Using proper isolation for the database
6. Optimization of triggers so the occurrence of deadlock can be minimized.

Based on our data model (ORION) which is a platform where students and professors from different universities can interact. The chances of deadlock occurring are minimal we are creating our database on acid. Atomicity, Consistency, Isolation, and durability. Using these chances of deadlock are greatly reduced

Meaning in our database all transaction will be committed, our data will be consistent, we have provided good isolation level hence no dirty read, and finally, it will be durable as the transaction will be saved.

Other than that we have provided several trigger optimization as well as query optimization so deadlock won't occur, we have also kept our transaction short and our design Is normalized.

Q4. Views & Synonyms

Views

Views can be defined as a virtual table which is created from the columns of one or more than one table, In other words, we can say that view is a logical concept we create to better queried the data which is used to display set column and is a virtual table.

Views can also be created from other views

The view can be said as a window from which access base table data

It doesn’t have its data all the data we see is from the base table or other views.

The view is stored in the database as a query in the data dictionary.

Synonym

A synonym is known as an alternate name or alias given to the database objects such as view, table, stored procedure, etc. In other words, we can say that synonyms mask the real name and owner of the object.

The main use of synonyms is to give the public access to the given object in the database without compromising its confidentiality. In oracle we can make synonym as both public and private, we can access different schema in the database without using a real schema name

Example view

We have two columns in the employment table employee name and salary so for see-through we can create a view of these two-column from the entire table. The benefit of view is that it hides the complexity of query and calculation.

Example synonym

Synonyms can be used by the company for security purposes so that users apply the query to synonyms rather than the base table itself.

Q.5 Querry processing phases

DBMS server has 3 general phases for query processing.

They are

1. Parsing
2. Execution
3. Fetching

Parsing

Parsing in this process SQL chooses the most efficient plan in which query can be executed. In other words, it can be said as query optimization phase. In this process, the query is broken down into a smaller unit and the original query is transformed into a version that is slightly different from the original code, the new code is generally more efficient than the previous one but the results are the same. In new code, the execution time of the query is reduced.

SQL uses a query optimizer which then analysis the given query and determines the most efficient path.

It validates several things such as syntax compliance, validating against data dictionary to crosscheck object names, it also checks if a user has proper security rights.

Then query optimizer analyzes and breaks into comparatively smaller components.

Finally, the query optimizer will select the most efficient execution plan.

1. SQL execution Phase

In the execution phase, all input-output operations are executed from the access plan, during the runtime of the access plan if the locks are required are used on the data to be accessed. Then data is retrieved from files and then is placed in a cache.

During the parsing and execution phase, all transaction management commands are processed by the DBMS to make sure any error won't occur.

1. SQL fetching Phase

When the above two phases are completed all the object that matches the given condition by the above phases are retrieved, grouped and sorted. In the fetching phase, the rows which are in the query result set are returned to the user. The DBMS store temporary data if required in the temporary tablespace. The cache is formed at both sides as server cache and client cache which is used to manage the flow of transaction of the data.