Pytania rekrutacyjne SQL

Teoria: często sie powtarzające podkreślone

**UNION/UNION ALL** - union pokazuje tylko unikatowe rekordy (wycina duplikaty), union all sklei wyniki 'jak leci', bez sprawdzania unikatowości -> union all zadziała szybciej

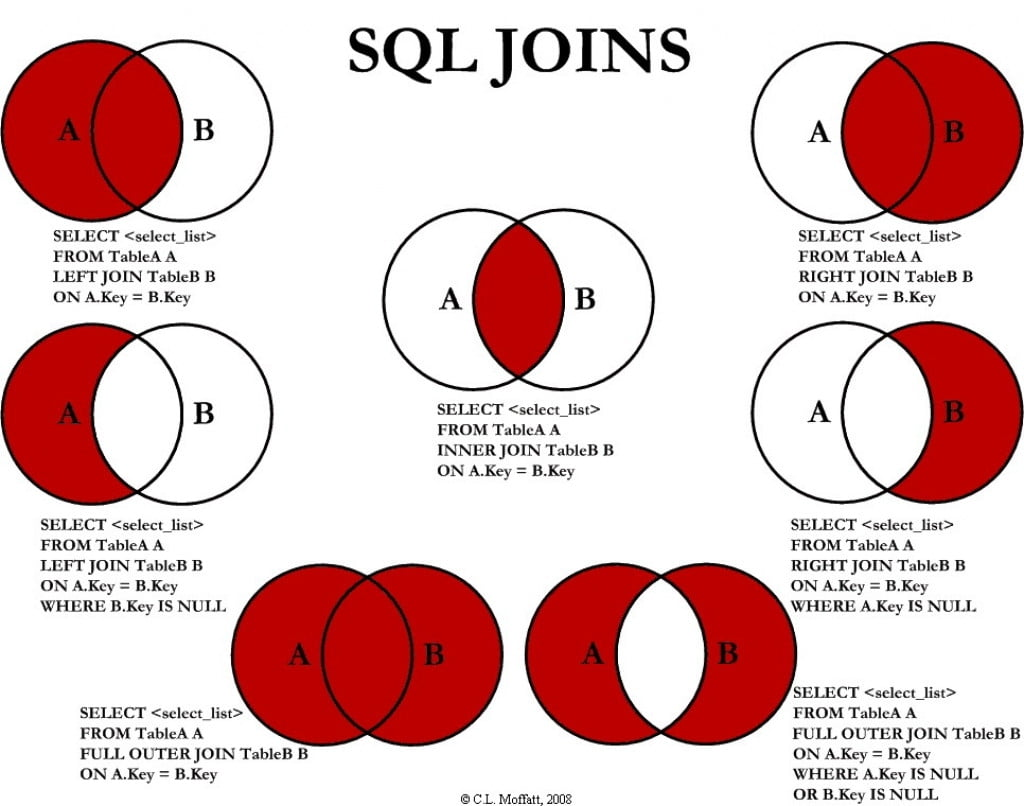
**DELETE/TRUNCATE/DROP** - sposoby usuwania danych:

**DELETE-**  usuwa wiersze w tabeli, może być ograniczony poprzez WHERE, JOIN, wedle fantazji (tak jak select) - usuwa wskazane wiersze (jeśli nie są wskazane - to wszytkie z tabeli)  
 delete usuwa wiersz po wierszu; jest odwracalny (rollback), commitowalny (tranzakcyjność)  
zostawia właściwości tabeli - constrainty, klucze, indexy

**TRUNCATE-**  czyści tabele - usuwa wszystkie dane na raz, szybszy niż delete. Zostawia tabele jakie obiekt, zostają klucze constrainty idexy itd.

Zasada działania – robi tak naprawdę drop + create table i zakłada indexy itd  
**DROP** - usuwamy tabele jakie obiekt z bazy, razem z indexami itd

**JOIN** – rodzaje (złączenia logiczne):

****

**JOIN (INNER JOIN)** - część wspólna A i B (rekordy występujące w Tabeli A i B)

rekordy występujące tylko w jednej z tych tabel - są pominięte w wyniku

**LEFT JOIN** - cała tabela A + to co do niej pasuje z tabeli B - tam gdzie można (jeśli jest spełniony warunek ON) dostawiamy dane z tabeli B, tam gdzie nie można (nie spełnia warunku ON) dostawiamy NULL

**FULL JOIN** - cała tabela A oraz cała tabela B, tam gdzie można to połączone

**CROSS JOIN** - wszystkie rekordy z A zestawione ze wszystkimi z B (‘przemnożone’ kazdy z kazdym)

**TRANSAKCJA, transakcyjność-** transakcja - zbiór operacji na bazie które wykonują się albo całkiem (wszystkie), albo w ogóle. transakcja musi być zcommitowana (commit/rollback)

**transakcja wewnątrz transakcji** (transakcje zagnieżdzone) - np przy robieniu ETLa, najpierw wstawiamy dane do tabel bazowych (“słowników’ z których korzystają inne tabele (poprzez klucz zewnętrzny))

**logi transakcyjne** - Log transakcyjny SQL Server’a zawiera rekordy, opisujące modyfikacje wykonywane na bazie. Dzięki tym informacjom istnieje możliwość odtworzenia bazy do określonego punktu w czasie, możliwość ponownego wykonania modyfikacji lub jej wycofania.

**ACID - *atomicity*, *consistency*, *isolation*, *durability -*** *zbiór właściwości gwarantujących poprawne przetwarzanie transakcji w bazach danych*

***A -*** *Niepodzielność transakcji oznacza, że każda transakcja albo zostanie wykonana w całości, albo w ogóle*

***C-****Spójność transakcji oznacza, że po wykonaniu transakcji system będzie spójny, czyli nie zostaną naruszone zasady integralności*

***I -*** *Izolacja transakcji oznacza, że jeśli dwie transakcje wykonują się współbieżnie, to zwykle (w zależności od poziomu izolacji) nie widzą wprowadzanych przez siebie zmian.*

***D -*** Trwałość danych oznacza, że system potrafi uruchomić się i udostępnić spójne, nienaruszone i aktualne dane zapisane w ramach zatwierdzonych transakcji, na przykład po nagłej awarii zasilania

**Prawo Demorgana**, teoria zbiorów

**LOGIKA TRÓJWARTOŚCIOWA** - TRUE, FALSE, NULL, różne zadanka na to

**Po co są joiny -** możemy uzsykać informacje będące w różnych tabelach, można użyć joina do ograniczenia wyników zapytania

**FUNKCJA/PROCEDURA -** funkcja ma argument (wartość w kolumnie, zmienna) i coś zwraca, procedura nie musi mieć argumentu (ale może, np tabele) oraz nie musi nic zwracać(może wysłać np maila);  
procedura może odpalać inne procedury i funkcje, funkcja może odpalać inne funkcje, ale nie procedury  
funkcje można użyć w selectie

**TYPY FUNKCJI -**tekstowe, agregujace, związane z data itd…   
skalarne (zwraca 1 wartość dla 1 zadanej) i wektorowe (zwraca wiele wartości dla 1 zadanej)

**PARTYCJONOWANIE -** podzielenie tabeli (dużej) na mniejsze częsci po jakiejś wartości (np jakieś okresy (W madrasie tmms\_ms\_resolution miał partycje po tpr\_id)), dane są przechowywane osobno wg partycji, usprawnia to działanie dużych tabel

**INDEXY** - index to obiekt na bazie przyspieszają odczyt danych z tabeli, ale spowalniają wprowadzanie danych, mogą zabierać dodatkowe miejsce (nieklastrowane)

**Klastrowany(clustered) -** to właściwie fizyczne posortowanie danych w tabeli wg zadanej wartości (1 lub wielu), tabela może mieć tylko 1 klastrowany index, index klastrowany nie zabiera dodatkowego miejsca

**Nieklastrowany(nonclustered) -** dodatkowy obiekt ‘tabela’ z rekordami posortowanymi wg zadanego klucza, może być ich wiele na jednej tabeli, każdy zabiera dodatkowe miejsce na bazie

**Columnstore –** wykorzystywany w hurtowniach danych

**KLUCZ(KEY) -** ograniczenia na table, mające na celu zapewnienie odpowiedniej postaci normalnej bazy. żeby klucz się założył dane w tabeli muszą być z nim zgodne.

“The Key, the whole Key, and nothing but the Key, so help me Codd”

**GŁÓWNY(PRIMARY) -** 1 na tabele, zapewnia unikatowość danych wg zadanej wartości, założenie klucza głównego powoduje stworzenie klastrowanego indexu na danych wartościach

**ZEWNĘTRZNY(OBCY,FOREIGN) -** ograniczenie wg którego dana wartość może zostać wprowadzona do tabeli tylko jeśli istnieje w wskazanej tabeli bazowej (‘słownik’). Jeśli tabela jest dla innych tabel bazową - to nie można z niej tak sobie usunąć danych, trzeba to robić od dołu (lub użyć w kluczu opcji DELETE ON CACSADE)

**CONSTRAINT** - ograniczenie danych w kolumnie, wg fantazji, np unikatowe, klucz zewnętrzny, wartości 1 2 3, itd, najczęściej constraintem jest klucz

**TABELA/WIDOK(VIEW) -** tabela - obiekt bazodanowy, w którym siedzą dane - są zapisane na stałe; widok to ‘wirtualna tabela’(zbudowana na bazie innych tabel/widoków)- wynik jakiegoś zapytania , odświeżany za każdym odwołaniem się do widoku - korzystanie z widoków jest obciążające pod względem obliczeniowym.

Widok – obciążenie obliczeniowe, tabela – obciążenie dyskowe

Szybszy odczyt z tabeli, szczególnie z odpowiednimi indexami

Teoretycznie i tabela i widok może mieć indexy

**WIDOK Z order by -** zasadniczo nie można, wyjątek - można zrobić, jeśli do utworzenia widoku uzyjemy top 100%

**modyfikacja danych poprzez widok -** możliwa w przypadku ‘prostych widoków’ (np. które są select \* z jednej tabeli, z dodanym warunkiem where)

**TABELA TYMCZASOWA/CTE(COMMON TABLE EXPRESSION) -** tabele CTE to zasadniczo widoki, dodatkowo nie można na nich zakładać indexów, CTE istnieje tylko kiedy jest wykonywane - można się odwołać tylko raz, w CTE można korzystać z rekurencji. Tabela tymczasowa - to tabela przechowywana na bazie TempDB, może mieć indexy, można się do niej odwoływać wiele razy

CTE wygląda tak:

WITH tabela1 as

(select \* from rawdata)

,tabela2 as

(select \* from stores)

Select \*

From tabela1 t1

JOIN tabela2 t2 on t1.ssho\_id=t2.sho\_id

**CZY SELECT \* jest ok?**

Na potrzeby sprawdzenia czegoś, wyciągnięcia danych zrobienia analizy na szybko (np. z własnej tymczasowej tabeli)

W ujęciu programistycznym, czyli robienie ETL, procedur,zakładanie widoków, zagregwanyc htabel, itd, kody który może być odpalony przez coś innego jest absolutnie niedopuszczalne, wypisujemy wszystkie kolumny z których korzystamy w kodzie SQL

**WHERE/HAVING -** having to ograniczenie na dane już zagregowane w grupy; where to ograniczenie wierszy które są brane pod uwage  
Czyli najpierw srpawdzany jest where -> agregacja -> sprawdzany having

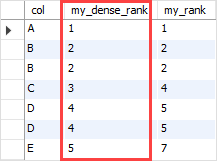
**CHAR/VARCHAR -** char ma określoną ilość znaków na stałe, w varchar określamy do jakiego zakresu (max); przy wyświetlaniu varchar pokaże po prostu wartość, char pokaże wartość i puste znaki (trzeba ztrimować)

**Jak wybrać wartość maksymalną bez funkcji max** - użyć top 1 i order by -

select top 1 wartość from table order by wartość desc

**Różnice coalesce(), isnull(X,Y)** - isnull sprawdza tylko jedną wartość czy jest NULL - argument X, jeśli nie jest Nullowa to wzraca X, jak jest to zwraca Y  
coalesce ma dowolną ilość argumentów - zwróci pierwszy podany który is not NULL

**RANK()/DENSE\_RANK() -** dense rank zawsze zwraca po kolei 1 2 3, rank w przypadku remisu pomija wartości



**OBSŁUGA BŁĘDÓW W SQL** – catch/try, raiseerror

**USUWANIE duplikatów z tabeli**- najprościej wybrać rekordy unikatowo z tabeli oryginalnej, wsadzić do tymczasowej, truncate oryginalnej, insert do oryginalnej

select distinct \*

into #temp

from oryginal

truncate table oryginal

insert into

oryginal

select \*

from #temp

**ALGORYTMY JOINÓW (ZŁĄCZENIA FIZYCZNE) – (pytanie bardziej na poziom mid/senior):**

**LOOP-NESTED JOIN –** zakłada 2 tabelki A,B. Bierze pierwszy rekord z A, szuka odpowiedników z B. jak znajdzie, to szuka następnego, wyczerpią się, to następny rekord z A (no 2 pętle jak w C++)

**MERGE JOIN**

**HASH JOIN -** najszybszy z nich, występuje tylko w przypadku EQUIJOINÓW (join na warunku =, żadne like czy between). Tworzy tabele hashy dla łączonych tabel i łączy je ze sobą. Tabela hashy – tabela fizycznych adresów komórek z danymi

**STATYSTYKI -** informacja o rozkładzie danych w danej tabeli, działają na podstawie indexów, sql server korzysta z tego w celu wykonania zapytania

**PLAN WYKONANIA ZAPYTANIA -** do analizowania wydajności

ctrl+L – plan estymowany (SSMS), można przeanalizować i na podstawie tego przemodelować zapytanie, zobaczyć co zjada ile zasobów, co się robi po kolei, jakie operatory są wykorzystywane

**OPTYMALIZACJA ZAPYTAŃ** - korzystamy z różnych sztuczek - indeksy nieklastrowany, statystyki, patrzymy w plan wykonania zapytania żeby usprawnić działanie query. Trzeba popatrzeć w plan zapytania na typy operatoró i złaczeń fizycznych jakimi rzuca mssql i spróbować tam coś nabajerować

**LOCK** - blokowanie danych na których pracuje procedura, zależnie od potrzeb na różnych poziowam

**NOLOCK** - przy selectie, odczyta dane niezależnie od aktualnych blokad (dane w trakcie modyhfikacji)

**DIRTY READS** - przeczytanie przez jedną transakcje danych jeszcze niskomitowanych w drugiej, zapobiega temu odpiwednie korzystanie z locków

**PROBLEM**: mając Oraclową bazę tranzakcyjną OLTP i data warehouse MSSQL, oraz potrzebę biznesową na zrobieniu raportu/dashboardu, omów proces optymalnego dostarczenia danego raportu (identyfikacja potrzeby biznesowej, jakie dane są nam potrzebne, identyfikacja czy dane siedzą w hurtowni, jeśli nie, to trzeba zrobić ETLa żeby siedziały, potem przygotować na warehousie obiekt z odpowiednio zebranymi,zagregowanymi danymi z którego raport wyciągnie dane w prosty sposób)

**OLAP/OLTP:**

OLTP – Online Transaction Processing – baza tranzakcyjna, na której jacyś operatorzy wykonują jakieś operacje. Będzie znormalizowana do 3NF albo dalej, wiele tabeli słownikowych itd., szybki insert

OLTP – Online Analytical Processing – bazy pod analitykę, czyli data science, reporting (BI) itd. Optymalizowane pod select, dużo indexów, często w postaci hurtowni – schemat gwiazdy (tabele faktór i wymiarów) lub snowflake

**TABELE FAKTÓW/WYMIARÓW** – w hurtowniach danych, w tabeli faktów mamy informacje że było zdazenie, z tabeli wymiarów dociągane jest info, np. o użytkowniku (1 wymiar), o lokacji (2 wymiar), o walucie( 3 wymiar) itd. Tabele wymiarów mają klucze Primary Key, a tabela faktów ma Foreign Key odnoszące się do wymiarów

**DENORMALIZACJA danych** – może zwiększyć wydajność zapytań select, kosztem zwiększonego zużycia dysku, tracimy też integralność danych (bo wszystko trzeba updatować osobnom, zamiast tylko tablicy słownikowej)

**Jak skopiować tebele** –

select \*

into kopia

from ….

**SNOWFLAKE dlaczego ciekawy** – Onlinowy datawarehouse, fajnie wpsółpracuje z AWS/AZURE (snow pipes które same mogą wgrywać dane do hurtowni na podstawie alertu jeśli plik wyląduje na blob storage), dużo funkcji, ciągle rozwijanych, porządna dokumentacja, opcja ‘time travel’ (cofnięcie się do tabeli sprzed jakiegoś wydarzenia (np. możemy znaleźć ID wydarzenia jakiegoś typu update, i oczytać dane z przed, np. przywrócić, nie trzeba analizować logów transakcyjnych), można sobie coś zundropować itd),

**jakie znasz schematy baz OLAP i róznice miedzy nimi**

-schemat gwiazdy : prostszy merytorycznie, mniej znormalizowany, lepszy performance selectów kosztem większego zużycia miejsca na dysku

-schemat snowflake(płatka śniegu) : trudniejszy (co ma znaczenie jeśli korzystają z hurtowni biznesowi analitycy), bardziej znormalizowany, gorszy performance selectu ale mniejsze zużycie miejcsa

**pokaż takie z A które nie są w B**

**left join:**

select a.\*

from a

left join b on a.id=b.id

where b.id is null

**subquery:**

select \*

from a

where a.id not in (select id from b)

**expect:**

select \* from a

except

select \* from b

tu najprostszy przykład subqury, moze być exist (jest dobry przy sprawdzaniu pozytywnym),

**Jak zrobić, żeby w raporcie użytkownik A widział tylko dane dla firmy X a użytkownik B tylko dla firmy Y?** odpowiedź:row level security

**różnice, plusy minusy oracle vs mssql:**

Oracle case sensitive, MSSQL nie

standarodowo autocommit on na MSSQL, off na Oracle

Oracle częściej wykorzystywany przy OLTP

T-SQL (MSSQL) prostszy dla użytkownika od PL SQL

MSSQL ma wersje popularną chmurową wersje co jest trendem rynkowym (Azure), Oracle tak nei bardzo (niby jest ale mało firm to ma)

**różnice, plusy minusy PowerBI vs Tableu:**

PBI prostszy

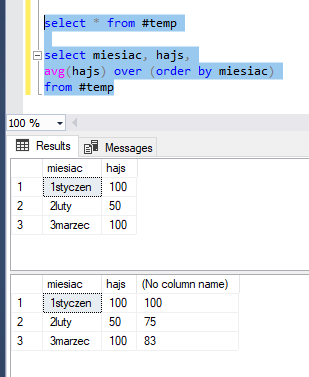
PBI dekstop darmowy

jako produkt microsoftu jest PBI prościej spiąć z innymi produktami (Excel, MSSQL, Azure)

Tableuo zadaje tragiczne(!!!) zapytania do bazy i ją obciąża

**ETL/ELT** – różnica polega na tych gdzie odbywa się transformacja danych – w ELT już po stronie hurtowni

**Running average** - trzeba użyć over , jak na screenie



**DDL/DML** -DDL – data definiition languege, CREATE, ALTER, DROP (table)

DML – data manipulation language, INSERT, UPDATE, DELETE

Select – dyskusyjny, można uznać za DML albo DQL (data quering language)

**Kolejność wykonywania zapytania (MSSQL):**

1. From
2. Where
3. Group by
4. Having
5. Select
6. Order by
7. Top

Select \* from tabela where 1=0 – zawsze zwróci 0 wierzy, bo warunek jest nie prawdziwy.

Pytanie jak zrobić w jednej linii tak, żeby był 1 wiersz?

Odp : -pogrupować coś, zagregować

-Wykomentować where, dać select top(1) \*

**Jak bezpiecznie przetrzymywać hasła użytkownika na serwerze?**

Nie przetrzymywać ich bezspośrednio, a wynik funkcji hashującej dla hasła i wpsrawdzać ten wynik w module logowania do aplikacji

**PYTANIA OD GURU SQL** – poziom zdecydowanie mid+/senior

1. Jakie są najlepsze praktyki przy konfiguracji bazy tempdb?
2. Czym jest wysoka dostępność i jak ją można osiągnąć w SQL Server?
3. Wymień 3 najważniejsze korzyści, jakie uzyskujemy z edycją Enterprise względem Standard?
4. Wymień rodzaje fizyczne i logiczne złączeń? Opisz jeden wybrany logiczny i jeden fizyczny.
5. Jakie znasz metody szyfrowania danych dostępne w SQL Server?
6. Jak działa operator key lookup? Jak optymalizować zapytania z takim operatorem?
7. Co to jest pierwsza postać normalna? Czym grozi jej nie przestrzeganie?
8. Co to jest deadlock? Jak go zdiagnozować? Jak sobie z nim radzić?
9. Jak poradzić sobie z problemem blokowania piszących przez czytających?
10. Jak pozbyć się własności identity z kolumny tabeli, w której już mamy dane?
11. Masz katalog, w którym znajduje się pełny backup bazy i backupy logów (bez zachowania konwencji nazewniczej). W jaki sposób odtworzyć automatycznie bazę ze wszystkimi logami do wybranego momentu w czasie?
12. Dostajesz błąd o braku miejsca na w logu transakcyjnym. Baza należy Always On Availability Group, jak będziesz postępować?
13. Otrzymujesz informację: „baza wolno działa”. Co sprawdzisz aby rozwiązać problem?

4.  
złączenia logiczne -:  
złączenie wewnętrzne (INNER JOIN)  
złączenie pełne (FULL JOIN)  
złączenie krzyżowe (CROSS JOIN)  
złączenie zewnętrzne lewo/prawo strone (LEFT/RIGHT JOIN)  
  
A  
LEFT JOIN B on [A.ID](http://a.id/)=[B.ID](http://b.id/)  
- w wyniku takiego złączenia otrzymamy wszystkie wartości z tabeli z tabeli A spełniające warunki zapytania oraz informację o  
tych wartościach z tabeli B, tam gdzie spełniony jest warunek złączenia (występują te wartości w tabeli B). Dla wierszy z A które nie mają odpowiednika z B - wyświetlone będą wartości NULL  
  
Wiersze z tabeli A które nie mają odpowiednika z tabeli B zostaną wyświetlone (w przeciwieństwie do INNER JOIN gdzie zostały by pominięte)  
  
złączenia fizyczne:  
Nested loop join  
Merge join  
Hash join  
  
Nested loop join - najprostszy algorytm złączeń, może być przyczyną słabej wydajności przy dużej ilości danych,  
może występować przy innych niż equi-join( znak równości =).  
Algorytm sprawdza wartości w pętli dla pierwszej z tabel - szuka wartości spełniających warunek złączenia z tabeli drugiej,  
po wyczerpaniu takich wartości przechodzi do następnego argumentu w tabeli pierwszej - do końca wartości  
  
7.  
Pierwsza postać normalna, że wiersze w tabeli muszą być unikatowe (względem klucza), a kolumnach w wierszu niepodzielne.  
Unikatowość danych w tabeli można uzyskać poprzez zastosowanie klucza głównego. Nieprzestrzeganie pierwszej postaci normalnej grozi:  
-pojawieniem się duplikatów danych, co utrudnia ich odczyt (np poprzez konieczność zastosowania distinct, który wymaga więcej obliczeń)  
-występowaniem danych, które nie są dostatecznie niepodzielne, więc trudniejsze w odczycie, wymagające użycia funkcji w celu uzyskania zadowalającego wyniku zapytania  
  
9.  
Z problem można poradzić sobie poprzez zmianę poziomu izolacji bazy ze standardowego READ COMMITED na inny.  
Jednak zmiana poziomu izolacji ma również inne konsekwencje, np READ UNCOMMITED sprawi, że czytający zobaczy dane jeszcze nie zcommitowane,  
READ COMMITED SNAPSHOT zwiększy obciążenie bazy tempdb  
  
10.  
poprzez  
ALTER TABLE (tabela)  
DROP COLUMN (kolumna)  
  
jeśli chcemy zachować dane z kolumny identity, ale bez opcji autoinkrementacji to:  
zakładamy nową kolumnę, kopiujemy dane z kolumny identity, usuwamy oryginalną kolumnę identity, zmieniamy nazwę nowej kolumny

I komentarze do nich:

Pytanie 4.

Świetnie. W odpowiedzi zawarłeś wszystko, czego oczekiwałem. Prawidłowo wymieniłeś wszystkie najważniejsze rodzaje złączeń, zarówno logiczne jak i fizyczne i opisałeś wybrane. Krótko, zwięźle i w samo sedno.

Pytanie 7

Jest bardzo dobrze. Zarówno wskazanie, na jakie wymagania stawia pierwsza postać normalna jest w porządku, jak i konsekwencje jej niestosowania. Zabrakło mi warunku o grupach zwielokrotnionych, choć widzę, ze przez większość teoretyków został on już odseparowany od pierwszej postaci normalnej. Zatem jest jak najbardziej ok.

Pytanie 9

Bardzo dobra odpowiedź – prawidłowo wskazałeś zmianę trybu izolacji transakcji na read uncommitted, jako jedno z możliwych rozwiązań i dobrze opisałeś konsekwencje jego zastosowania. Wykazałeś się znajomością opcji READ COMMITTED SNAPSHOT. Warto wiedzieć, że mamy jeszcze parę opcji:

1. Przy Always on Awailability Groups – odczyt z dodatkowej repliki przeznaczonej do odczytu.
2. Utworzenie snapshota bazy danych – jeśli godzimy się na odczyt nie do końca aktualnych danych. Aktualność na moment wykonania snapshota.
3. Log shipping + pozostawienie bazy odtwarzanej w trybie STANDBY (stan na moment odtworzenia ostatniego backupu loga)

Pytanie 10

Metoda, którą opisałeś rzeczywiście zadziała, ale jest bardzo ciężka. Wymaga fizycznego update’u wszystkich danych i zaburza kolejność kolumn. Dla dużej tabeli nie zrobimy tego bez downtime’u. Jest dużo lepsza metoda, która pozwala na przeprowadzenie operacji bez przesunięcia jakiegokolwiek rekordu. Polega na tym, że tworzymy obok, na tej samej filegroupie tabelę o identycznej strukturze i jakich samych indeksach, a następnie wykonujemy

1. ALTER TABLE <stara> SWITCH TO <nowa>
2. Sp\_rename <stara>,<tymczasowa>
3. Sp\_rename <nowa>,<stara>

Jeśli nie było na tabeli kluczy obcych i żadne na nią nie wskazywały, operację daje się przeprowadzić w kilka milisekund, niezależnie od wielkości tabeli. Zdaję sobie sprawę, że nie jest to powszechna wiedza. To było zadanie „z gwiazdką”, wielkie dzięki za podjęcie wyzwania. Bynajmniej nie znaczy to, że nie zaliczam Twojej odpowiedzi. Zanim poznałem tę metodę robiłem to tak, jak to opisałeś.

nieopisane pytanka:

SQL lub PL/SQL IDE - jakie znasz, z jakiego korzystasz + wymień skróty klawiszowe z których korzystasz

Dwa typy tabel w schemacie gwiazdy + jakie typy danych przechowują

Wymień znane ci RDMS

Zalety i wady noSQL

Dodawanie liczb binarnych

W jakimkolwiek języku programowania napisz skrypt, który:

stworzy folder

stworzy plik i wrzuci go do tego folderu

stworzy subfolder

skopiuje plik i wrzuci do subfolderu

Co to jest data mart?

FOR a WHILE - różnice

Co to jest rekurencja (żeby zrozumieć rekurencję, musisz najpierw zrozumieć rekurencję ☺ )

Hurtownia danych a baza transakcyjna

Opisać 2 algorytmy sortowania danych (pamiętam, że odpisałem na to pytanie: "Pewnie nie chodzi o ascending/descending? :) " (Uśmieszek jak w oryginale)

Model tabel dla wypożyczalni aut i 3 scenariusze testowe podać przykładowe

Opisać znane constrainty

Obiekt a klasa

Jak się modeluje związki wiele do wiele

W obojętnie jakim języku programowania (nawet to nie musi być język, byle składnia jako-taka i słowa klucze), napisz skrypt, który

- posortuje (nie pamiętam rosnąco czy malejąco) elementy wektora

- bez używania funkcji minimum, maksimum, sort

Co to jest usługa on permise i czemu jest coraz bardziej popularna

Model tabel dla sklepu internetowego (tak, żeby było info o zamówieniu, produktach, klientach)

Alter table modify, alter table add constraint

DDL a DML

Jakimi poleceniami nadaje się uprawnienia

Napisz selecta/funkcję/procedurę, która wypluje obecną datę i godzinę w formacie 24h (a nie 12h)

39. Wady indeksów