Teoretická/vyhledávací část – zadání:

! Úkoly jsou koncipovány, tak abyste odpovědi hledali a získali tak trochu přehled, nejedná se o vědomostní test.

Nalezené výsledky vkládejte do libovolného textového dokumentu do složky “Odpovědi”.

Zde netřeba uvádět postup, kde a jak jste informaci našli.

1. Uveďte nukleotidovou sekvenci exonu 9 genu CFTR. (tip: hledejte “ensembl”)

2. Uveďte, jaké onemocnění způsobují patogenní mutace v genu CFTR a některé uveďte.

3. Uveďte, co znamená autosomálně recesivní přenašečtví.

4. Rozepiště, jak byste postupovali při vytváření SQL databáze na Linux serveru, tak aby byla zajištěna integrita dat.

5. Uveďte, co znamená *SQL* Injection a jak se tomu vyvarovat.

6. Zjistěte, co znamená error: “Error in .local: Cannot allocate a new connection: 16 connections already opened” a napiště jak byste postupovali při jeho opravě.

7. Zjistěte, co znamená error: “Error in if: argument is of length zero” a napiště jak byste postupovali při jeho opravě.

8. Napište, proč se dělá Sekvenování nové generace (NGS)?

Teoretická část – odpovědi:

1. Uveďte nukleotidovou sekvenci exonu 9 genu CFTR. (tip: hledejte “ensembl”)

Správná odpověď:

přepis sekvence 1:
GGATTTGGGGAATTATTTGAGAAAGCAAAACAAAACAATAACAATAGAAAAACTTCTAAT
GGTGATGACAGCCTCTTCTTCAGTAATTTCTCACTTCTTGGTACTCCTGTCCTGAAAGAT
ATTAATTTCAAGATAGAAAGAGGACAGTTGTTGGCGGTTGCTGGATCCACTGGAGCAGGC
AAG

Přepis sekvence 2:
GGATTTGGGGAATTATTTGAGAAAGCAAAACAAAACAATAACAATAGAAAAACTTCTAAT

GGTGATGACAGCCTCTTCTTCAGTAATTTCTCACTTCTTGGTACTCCTGTCCTGAAAGAT

ATTAATTTCAAGATAGAAAGAGGACAGTTGTTGGCGGTTGCTGGATCCACTGGAGCAGGC

AAG

Postup:

- najít si lidský gen CFTR:

https://www.ensembl.org/Human/Search/Results?q=cftr;site=ensembl;facet_species=Human

- hledat transkript genu a přímo 9. exon: ENSE00003976986

https://www.ensembl.org/Homo_sapiens/Transcript/Exons?db=core;g=ENSG00000001626;r=7:117480025-117668665;t=ENST00000003084

- potvrdit si v BLAST, že se opravdu jedná o 9. exon genu CFTR

https://blast.ncbi.nlm.nih.gov/Blast.cgi

- původně jsem našel v databázi ensembl tuto sekvenci lidského genu CFTR. Ale když jsem sekvenci kontroloval v BLAST, tak mě to upozornilo, že se jedná o 8. exon (nikoliv 9. exon) genu.

- 8.exon genu cftr:
GATTTCTTACAAAAGCAAGAATATAAGACATTGGAATATAACTTAACGACTACAGAAGTA
GTGATGGAGAATGTAACAGCCTTCTGGGAGGAG

- najít si lidský gen CFTR:
https://www.ensembl.org/Human/Search/Results?q=cftr;site=ensembl;facet_species=Human

- hledat transkript genu (8.exon cftr: ENSE00000718637)
https://www.ensembl.org/Homo_sapiens/Transcript/Exons?db=core;g=ENSG00000001626;r=7:117480025-117668665;t=ENST00000003084

- ověřit si sekvenci v nástroji BLAST → jedná se o 8.exon:
https://blast.ncbi.nlm.nih.gov/Blast.cgi

Při zadání sekvence do BLAST mi ale nástroj potvrdil, že daná sekvence má 100% shodu s 8. exonem lidského genu CFTR. Atak je potřeba hledat znovu a zadat opravdu 9. exon (viz níže).

2. Uveďte, jaké onemocnění způsobují patogenní mutace v genu CFTR a některé uveďte.

Patogenní mutace v genu CFTR (Cystic Fibrosis Transmembrane Conductance Regulator) způsobují onemocnění nazvané cystická fibróza (CF). Cystická fibróza je genetické onemocnění, které postihuje převážně dýchací a trávicí systémy. Zde jsou některé příznaky a projevy cystické fibrózy:

1) Problémy s dýcháním:

- Opakované infekce dýchacích cest
- Obtíže s dýcháním a kašel
- Chronická bronchitida

2) Problémy s trávením a výživou:

- Obtíže s trávením a absorpcí živin:
- Nízká hmotnost nebo nárůst hmotnosti
- Přetrvávající průjmy

3) Problémy s pankreatickou funkcí:

- Nedostatečná produkce pankreatických enzymů
- Nedostatečná produkce inzulínu

4) Sůl v potu:

- Lidé s cystickou fibrózou mají zvýšenou koncentraci soli (chloridu) ve svém potu.

5) Problémy s reprodukčním systémem:

-Někteří muži mohou mít problémy s plodností způsobené absencí chámovodu (*vas deferens*).

Cystická fibróza je způsobena mutacemi v genu CFTR, který reguluje transport chloridových iontů přes buněčné membrány. Tyto mutace vedou k tvorbě hustého a lepivého hlenu, což vede k opakovaným infekcím a postižení orgánů, jako jsou plíce, pankreas a trávicí trakt.

Je důležité si uvědomit, že existuje mnoho různých mutací v genu CFTR, a to i mezi lidmi s cystickou fibrózou. Různé mutace mohou způsobit různé stupně závažnosti onemocnění.

Odkazy:
https://cs.wikipedia.org/wiki/Cystick%C3%A1_fibr%C3%B3za

https://www.repromeda.cz/slovnicek-pojmu/cysticka-fibroza/

https://klubcf.cz/o-cysticke-fibroze/o-nemoci/co-je-cysticka-fibroza/

3. Uveďte, co znamená autosomálně recesivní přenašečství.

Autosomálně recesivní dědičnost (AR) se týká genů umístěných na nepohlavních chromozomech – autosomech. Sledujeme přenos znaku podmíněného recesivní alelou. Fenotypově se sledovaný znak projeví pouze u recesivních homozygotů (aa). Heterozygoti (Aa) jsou bez klinického projevu, nelze je fenotypově odlišit od dominantních homozygotů (AA). Nicméně s 50% pravděpodobností předají svým potomkům recesivní alelu, tzn. mohou být přenašeči autozomálně recesivního znaku/ onemocnění. Aby se u potomka nemoc projevila, je nutné, aby mutovanou – recesivní alelu předal i druhý rodič (který tím pádem musí být také minimálně přenašeč). Pokud máme dva rodiče, oba heterozygoty – přenašeče – poté má každý z nich právě 50% šanci, že předá onu mutovanou – recesivní alelu. Těmto rodičům se (podle teoretické pravděpodobnosti) narodí zdravé dítě (dominantní homozygot) ve 25% případů, přenašeč (heterozygot) v 50% případů a nemocné dítě (recesivní homozygot) ve 25% případů.

Odkazy:
http://www.vrozene-vady.cz/genetika/index.php?co=dedicnost

https://www.wikiskripta.eu/w/Autosom%C3%A1ln%C4%9B_recesivn%C3%AD_d%C4%9Bdi%C4%8Dnost

4. Rozepište, jak byste postupovali při vytváření SQL databáze na Linux serveru, tak aby byla zajištěna integrita dat.

Při vytváření SQL databáze na Linux serveru s důrazem na zajištění integrity dat můžete postupovat následovně:

1) Instalace Databázového Systému:

- Nainstalujte vybraný databázový systém pomocí správce balíčků podle vaší linuxové distribuce.

2) Přihlášení a Vytvoření Databáze:

- Přihlaste se do databázového systému a vytvořte novou databázi.

3) Definice Schématu:

- Nadefinujte strukturu tabulek vytvořením potřebných sloupců, primárních a cizích klíčů.

4) Vytvoření Indexů:

- Přidávejte indexy na sloupce, které budou často používány ve vyhledávání.

5) Použití Trigerů pro Integritu Dat:

- Vytvořte triggery, pokud potřebujete implementovat specifická pravidla integrity dat.

6) Použití Transakcí:

- Manipulujte s daty v rámci transakcí pro zajištění atomickosti operací.

7) Zálohování Databáze:

- Pravidelně provádějte zálohy databáze pro obnovení dat v případě potřeby.

8) Monitorování a Logování:

- Monitorujte výkon a využívejte logování pro sledování operací a odhalení problémů.

Toto jsou obecné kroky a pro každý z nich budete potřebovat odpovídající příkazy specifické pro vámi zvolený databázový systém. Příkazy se mohou lišit napříč různými systémy (PostgreSQL, MySQL, SQLite atd.), a proto je důležité konzultovat dokumentaci konkrétního systému pro přesné příkazy.

5. Uveďte, co znamená *SQL* Injection a jak se tomu vyvarovat.

SQL Injection je druh útoku na webovou aplikaci, při kterém útočník vkládá nebo injektuje škodlivý SQL kód do SQL dotazu, který je následně prováděn na databázovém serveru. Cílem tohoto útoku je získání neautorizovaného přístupu k databázovým informacím nebo narušení integrity dat. Těmito způsoby můžete minimalizovat riziko SQL Injection a zabezpečit vaši webovou aplikaci proti tomuto typu útoku.

Postup k zabránění SQL Injection:

1) Používání parametrizovaných dotazů:

- Nepřímo vkládejte uživatelské vstupy do SQL dotazů. Místo toho používejte parametrizované dotazy nebo předem připravené dotazy, které oddělují data od příkazu.

2) Ověření a validace uživatelských vstupů:

- Ověřujte a validujte vstupy od uživatelů. Ujistěte se, že přijatá data odpovídají očekávaným formátům a rozsahům.

3) Omezení oprávnění databázových účtů:

- Databázovým účtům přidělte minimální nutná oprávnění pro vykonávání operací. Nepoužívejte účty s vysokými oprávněními pro běžné operace.

4) Používání bezpečných api a frameworků:

- Pokud je to možné, používejte bezpečné API a frameworky, které mají zabudované ochrany proti SQL Injection. Moderní frameworky často nabízejí funkce pro automatickou ochranu.

5) Šifrování dat a komunikace:

- Používejte šifrování pro citlivá data, jako jsou hesla a citlivé informace. Také zabezpečte komunikaci mezi webovou aplikací a databázovým serverem.

6) Přístup k databázi z principu nejmenšího oprávnění:

- Udělte databázovým účtům pouze ta oprávnění, která jsou nezbytně nutná pro jejich funkci. Omezte jejich schopnost manipulovat s citlivými daty.

7) Monitorování a logování:

- Monitorujte logy aplikace a databáze pro detekci neobvyklého chování nebo podezřelých dotazů. Aktivně sledujte a reagujte na možné útoky.

8) Aktualizace a záplaty:

- Pravidelně aktualizujte a záplatujte váš software, včetně databázových systémů a aplikačního kódu, aby byly odstraněny potenciální bezpečnostní chyby.

6) Zjistěte, co znamená error: “Error in .local: Cannot allocate a new connection: 16 connections already opened” a napiště jak byste postupovali při jeho opravě.

Tento druh chyby "Error in .local: Cannot allocate a new connection: 16 connections already opened" se obvykle objevuje v kontextu práce s databázovými připojeními, a to zejména v případě, kdy je dosaženo limitu maximálního počtu otevřených spojení s databází. Oprava této chyby může vyžadovat kombinaci těchto kroků, v závislosti na konkrétních okolnostech a architektuře vaší aplikace.

Chybová zpráva indikuje, že aplikace nebo proces již otevřel 16 spojení s databází a není schopen vytvořit další.

Postup k opravě této chyby může být následující:

1) Zkontrolujte počet otevřených připojení:
- Zjistěte, kde v kódu nebo konfiguraci se vytvářejí nová připojení k databázi. Ověřte, zda jsou správně uvolňována a zavírána po použití.

2) Použijte connection pooling:
- Místo vytváření nových připojení pro každý dotaz můžete použít techniku známou jako connection pooling. Connection pooling umožňuje znovupoužívání existujících připojení, což může snížit počet otevřených připojení.

3) Zvyšte limit připojení:
- Pokud je to možné a odpovídá to konkrétním požadavkům databázového serveru, můžete zvýšit limit maximálního počtu otevřených připojení. Toto lze udělat v konfiguraci databázového serveru.

4) Optimalizujte kód:
- Zkontrolujte, zda váš kód efektivně využívá připojení k databázi. Optimalizujte dotazy a ujistěte se, že připojení jsou otevřena a uzavřena v optimálních bodech.

5) Zamyslete se nad architekturou aplikace:
- Přemýšlejte o celkové architektuře vaší aplikace a jakým způsobem pracuje s databází. Zvažte možnost použití centralizovanějšího přístupu k připojením, například prostřednictvím správce připojení nebo služby pro řízení stavu.

6) Zkontrolujte logy databázového serveru:
- Podívejte se do logů databázového serveru, aby bylo možné identifikovat, kde a proč jsou vytvářena nová připojení. To vám může poskytnout užitečné informace pro opravu chyby.

7) Zvažte horizontální škálování:
- Pokud se jedná o velkou a intenzivně vytíženou aplikaci, může být vhodné zvážit horizontální škálování, tj. distribuci zátěže mezi více databázových serverů.

7. Zjistěte, co znamená error: “Error in if: argument is of length zero” a napiště jak byste postupovali při jeho opravě.

Chyba "Error in if: argument is of length zero" v jazyce R obvykle znamená, že v podmínce (**if**) je použitý výraz, který nemá žádnou délku. To může nastat, pokud se pokoušíte pracovat s prázdným objektem nebo proměnnou.

Těmito postupy můžete minimalizovat riziko chyby "Error in if: argument is of length zero".

Postup pro opravu této chyby by mohl být následující:

1. **Zkontrolujte Vstupy:**
    - Ujistěte se, že proměnné nebo objekty, které používáte v podmínce, mají očekávanou strukturu a nejsou prázdné. Můžete použít funkce jako **length()**, **NROW()**, nebo **is.null()** k ověření délky nebo existence objektu.
2. **Přidání Podmínky na Délku Objektu:**
    - Před použitím podmínky (**if**) zkontrolujte délku objektu. Použití podmínky na délku může zabránit chybě, pokud objekt nemá žádnou délku.
3. **Ošetření Prázdných Objektů:**
    - Pokud očekáváte, že váš objekt může být prázdný, přidejte podmínky, které ošetřují tuto situaci.

8. Napište, proč se dělá Sekvenování nové generace (NGS)?

Sekvenování nové generace (Next-GenerationSequencing, NGS) se provádí z několika důvodů, a to především kvůli svým vysokým výkonům, rychlosti a schopnosti generovat obrovské množství genomických dat. Mezi hlavní důvody patří: výzkum genomu (identifikace genů, strukturálních změn DNA…), diagnostika nemocí (identifikace mutací a genetických variant v genomu pacienta, diagnostika predispozic k onemocnění), farmakogenomika (identifikace genetických variant pro předvídání účinnosti léků), výzkum rakoviny a další. NGS je klíčovým nástrojem v oblasti genetického výzkumu a diagnostiky, poskytujícím množství informací o genetickém kódu, které jsou klíčové pro pochopení života a léčbu genetických onemocnění. Tato metoda umožňuje sekvenovat úseky nukleových kyselin až celých genomů podstatně rychleji a cenově výhodněji než ostatní sekvenační metody. Podstatou je zpracovávání tisíců až milionů sekvencí současně v jednom běhu, což má za následek produkci obrovského množství výstupních dat.

Odkazy:

https://www.generi-biotech.com/cs/princip-ngs-metody/
https://www.wikiskripta.eu/w/Sekvenov%C3%A1n%C3%AD_DNA