

Feladatok automatikus értékelése adatbázis-kezelés oktatásában

EÖTVÖS LORÁND TUDOMÁNYEGYETEM

INFORMATIKAI KAR

MÉDIA- ÉS OKTATÁSINFORMATIKAI
TANSZÉK



Szerző:

Horcsin Bálint

programtervező informatikus BSc

3. évfolyam

Témavezető:

Dr. Abonyi-Tóth Andor

egyetemi docens

Budapest, 2024

Tartalomjegyzék

1. Absztrakt	3
2. Bevezetés	4
2.1. Témaválasztás indoklása, a dolgozat előzményei	4
2.2. Dolgozat felépítése	5
3. Szakmódszertani elvárások	6
3.1. Az Adatbázisok I tárgy felépítése	6
3.2. Kódminőségtől független értékelés	7
3.3. Visszajelzés típusa	7
3.4. Iparban alkalmazható nyelvek használata	8
3.5. Nem tökéletes megoldások értékelése	8
3.6. Oktatói felügyelet	9
4. Értékelési módszerek	10
4.1. Szakirodalmi áttekintés	10
4.1.1. Statikus kódanalízis (white-box tesztelési módszer)	11
4.1.2. Dinamikus ellenőrzés (black-box tesztelési stratégia)	12
4.1.3. Visszajelzések típusai	16
4.1.4. Beváltásvizsgálatok	17
4.2. Javasolt értékelési módszerünk	18
4.2.1. Biztonság	20
5. Felhasználói felületek	21
5.1. Hallgatói felület	21
5.2. Oktatói felület	24
6. Hatásvizsgálat	26
6.1. A javítás pontossága	27

6.2. Oktatói munka	28
6.3. Hallgatói vélemények	28
6.3.1. Alkalmazható próbák	29
6.3.2. Tárgyak nehézsége	30
6.3.3. Zárthelyi dolgozattal és értékelővel kapcsolatos vélemények	31
6.3.4. A hatásvizsgálat során nem vizsgált csoportok	34
6.4. Felmerülő problémák	34
6.5. Hatásvizsgálat összefoglalása	37
7. Összefoglalás	40
7.1. Javaslataink	40
7.2. Hozzájárulásunk	41
7.3. További kutatási irányok	41
Köszönetnyilvánítás	43
A. Hallgatóknak kiadott kérdőív	44
B. Mesterséges intelligencia használata	47
Irodalomjegyzék	48

1. fejezet

Absztrakt

Jelen kutatás célja olyan módszer keresése volt adatbáziskezelés-feladatok automatikus értékelésére, amely alkalmazható az ELTE Informatikai Karán a programtervező informatikus képzésben, illetve később az informatika (digitális kultúra) tanárszakon is. Célunk volt, hogy beválasztvizsgálat alapján javaslatot tegyünk az esetleges szélesebb körű bevezetésre, meghatározzunk rá feltételeket, valamint a potenciális veszélyforrásokat azonosítsuk. Az általunk kialakított módszerben a vonatkozó szakági szokások közül a black-box tesztelési stratégiát alkalmaztuk.

A kutatás eredményeképpen sikerült kialakítanunk egy olyan tesztelési módszert, amely különböző adatbáziskezelő-rendszerekre adaptálható.

A beválasztvizsgálat során kiderült, hogy a hallgatók az értékelőrendszer által előkészített, majd az oktatók által áttekintett osztályzatot igazságosnak tartották. A korábbi vonatkozó szakági normákkal ellentétben, az egyes feladatok ellenőrzésekor rendkívül nagy pontosságot tudtunk elérni még akkor is, hogyha egyetlen teszttel futtattuk le az értékelést. Ugyanakkor az értékelés pontossága növelhető, hogyha néhány típushibára is készítenek tesztet, de az értékelés pontossága nélkül is kielégítő. Ezáltal zárthelyi dolgozatok javítása során biztosan alkalmazható a rendszer, ha az értékelések egy gyors oktatói felülvizsgálaton is átesnek. Ezzel növelhető az oktatói munka hatékonysága.

A hallgatók igényelték, hogy a zárthelyi dolgozatot megelőzően a rendszerben lehessen gyakorolni egy mintaként készített zárthelyi dolgozattal, és hasznosnak tartották volna, hogyha lett volna otthoni gyakorlásra elérhető automata tesztelő.

Összességében megállapítottuk, hogy a hallgatók számára a zárthelyi dolgozatok megírásakor problémát a rendszer bevezetése nem okozott, az oktatói munka hatékonyságát jelentősen növelte. Ezáltal javasoljuk a rendszer szélesebb körű bevezetését.

2. fejezet

Bevezetés

2.1. Témaválasztás indoklása, a dolgozat előzményei

Adatbáziskezelés-feladatok automatikus értékelésére már évekkel ezelőtt felmerült az igény az ELTE IK Információs Rendszerek Tanszékén a Telekommunikációs hálózatok tárgy gyakorlatához hasonlóan, ugyanakkor az ehhez szükséges fejlesztési munkákat korábban nem végezték el. Ettől függetlenül a 2022/2023-as tanévben elkezdtünk dolgozni egy középiskolai oktatásban alkalmazható automata értékelőrendszeren.

Ennek a motivációja az volt, hogy az (Oktatási Hivatal, é. n.) közzéteszi a mindenkori érettségi eredményeket anonimizált formában, és az alapján az adatbázis-kezelés egy olyan területe volt az érettségi vizsgának, amelyben a tanulók teljesítménye nem volt elégséges. Ezt (Siegler, 2014) részletesen elemezte, és levonta következtetésként, hogy a legnagyobb nehézséget a programozás jelenti az érettségiben, de problémát jelent az adatbázis-kezelés is. A Nemzeti Alaptanterv 2020-as változása nyomán bevezetésre került az informatika tárgy helyett a digitális kultúra, amely következményeként középszintű érettségi vizsgán továbbra is használható a Microsoft Access szoftver, ugyanakkor emelt szinten MariaDB-t szükséges használni¹. Ez hívta életre (Király és mtsai., 2022) által bemutatott <https://sqlsuli.hu/> portált, ugyanakkor szerettünk volna egy általánosabb rendszert alkotni, amellyel a jövőben tetszőleges lekérdezést is fogunk tudni tesztelni. Egyszerűsíteni akartuk a középszint és az emelt szint közötti áttérést, így azt tartottuk célszerűnek, hogy Microsoft Access-t is és MySQL-t is támogasson rendszerünk.

¹A dolgozatban a MariaDB és MySQL elnevezéseket egymás szinonimájaként fogjuk használni, mivel jelen kézirat szempontjából a kettő ekvivalensnek tekinthető, mivel utasításkészletük lényegében megegyezik, a MariaDB és a MySQL közös gyökere valamint a kompatibilitásra való törekvés miatt.

Ez jelentős implementációs nehézséggel járt (például virtuális gépeket hatékonyan kellett tudnunk indítani). A problémákat megoldottuk, mely informatikatudományi és szakmódszertani részleteiről beszámoltunk egy TDK-dolgozatban 2023 tavaszán, de a dolgozat sikeréhez kellett volna készíteni hatásvizsgálatot is. A rendszer középiskolákban történő bevezetése és hatásvizsgálat végzése jelentős forrásokat igényelt volna, egyelőre ezt nem tartottuk megoldhatónak. (Horcsin és Abonyi-Tóth, 2023) munkánkban publikáltuk a középiskolai felhasználásra javasolt megoldásunkat, amelyet jelen dolgozatban tovább javítottunk.

Egyeztettünk az ELTE IK Információs Rendszerek Tanszékével egy ilyen jellegű rendszer egyetemen történő bevezetéséről. Bizonyos implementációs részletekben (az alkalmazott .Net Framework keretrendszer, MySQL támogatása) még mindig őrzi nyomát a megoldás, hogy a középiskolás diákok jelentették az eredeti célcsoportot. De mind az alkalmazott módszert, mind az implementációt modernizáltuk az ELTE tantárgypedagógiai igényeinek megfelelően (például a Microsoft Access támogatása elveszett, de lehetővé tettük zárthelyi dolgozatok kitűzését, valamint implementációnkat integráltuk az ELTE IK nyílt forráskódú automata tesztelőjével, a TMS rendszerrel²).

A jelenlegi hosszabb távú célunk az ELTE IK-n egy adatbáziskezelés-feladatok automatikus értékelésére alkalmas rendszer bevezetése, melyre a javasolt megoldásunkat jelen dolgozat mutatja be. Javaslatunk alátámasztására pilot rendszerben, egy szűk hallgatói körre gyakorolt hatását vizsgáltuk egy ilyen rendszernek, amelyhez hasonló vizsgálatról a hazai szakirodalom korlátozottan számolt eddig be.

2.2. Dolgozat felépítése

Jelen dolgozatban röviden áttekintjük az ELTE IK-n az Adatbázisok I tantárgy gyakorlatának tantárgypedagógiai elvárásait. Ezt követően áttekintjük a szakirodalom vonatkozó részeit, vizsgáljuk ezek alkalmazhatóságát az ELTE-n. Javasunk a korábbiaktól eltérő, új módszert, amely jobban illeszkedik az ELTE módszertanához.

Ezt követően bemutatjuk hatásvizsgálatunk eredményét, a vizsgálat során felmerülő problémákat. Majd pedig levonunk következtetéseket, és javaslatokat teszünk az oktatásban való hosszú távú bevezetésre.

²A TMS az ELTE IK-n működő webes felületen keresztül elérhető értékelőrendszer, amelybe a hallgatók fel tudnak tölteni programozási feladatokra megoldásokat, és automatikusan visszajelzést kaphatnak arról, hogy a beküldött programjuk helyes-e. („Task Management System”, é. n.)

3. fejezet

Szakmódszertani elvárások

Az értékelőrendszerrel kapcsolatban fontos kérdés volt, hogy az általa adott eredmény mennyire állna közel egy oktató által végzett értékeléshez. Ezáltal rögzíteni kellett azokat az elveket, amelyek mentén az ELTE IK-n értékelnek az Adatbázisok I gyakorlat oktatása során.

3.1. Az Adatbázisok I tárgy felépítése

Az ELTE IK programtervező informatikus szakára jellemző módon az Adatbázisok I tárgy oktatása két különálló tárgyként van adminisztrálva. Az egyik egy heti 2 órából álló gyakorlati tárgy (Adatbázisok I gyakorlat a neve), a másik pedig heti 2 előadásból álló tárgy (Adatbázisok I előadás a neve). Mindkettő 2 – 2 kreditet ér, a gyakorlat folyamatos számonkérésű¹ (heti kisebb számonkérések vannak – órai munkát fel kell tölteni –, és két nagy zárthelyi dolgozatot kell a hallgatóknak teljesíteniük), az előadás anyagából vizsgát kell tenniük a hallgatóknak.

A hallgatóknak a gyakorlaton a félév első felében elsősorban DQL² lekérdezéseket kell készíteniük Oracle Database használatával, valamint relációs algebrával és kiterjesztett relációs algebrával kapcsolatos feladatokat kell megoldaniuk papíron. A félév máso-

¹Az ELTE IK-n több tárgy is ún. *folyamatos számonkérésű*, vagyis a hallgatóknak heti-kétheti rendszerességgel kell tárgytól függően ún. *röpsz*-t írniuk a gyakorlat idejében, valamint házi feladatokat beadniuk. Ilyen tárgy többek között a *Matematikai alapok*, a *Programozás alapjai*, az *Imperatív programozás*, a *Funkcionális programozás*, az *Objektumelvű programozás*, az *Analízis I és II*. Ezen tárgyak oktatói jellemzően pozitívan értékelik, hogy heti szintű kisebb számonkéréssel a hallgatók rá vannak szoktatva a félév során történő folyamatos tanulásra.

²*Data Query Language*: adatlekérdező nyelv, vagyis SELECT lekérdezések

dik felében megismerkednek a hallgatók a relációs adatbázisok tervezésével és a DML³ lekérdezésekkel, valamint procedurális SQL kódot (PL/SQL) is írniuk kell.

Az első zárthelyi dolgozat során fele-fele arányban kell jellemzően papíros és számítógépes feladatokat megoldaniuk. A második zárthelyi dolgozatban elsősorban procedurális SQL kódot kell írniuk, kisebb részben pedig adatbázis-tervezési feladatokat kell megoldaniuk és DML utasításokat írniuk.

3.2. Kódminőségtől független értékelés

A feladatokat több, különböző stratégia mentén is meg lehet oldani. A zárthelyi dolgozatok áttekintése során kiderült, hogy olyan gondolatmeneteket követnek egyes hallgatók, amelyre az oktatók egyáltalán nem gondolnának, mégis helyes a megoldása a hallgatónak. A speciális gondolatmenet mindenképpen üdvözlendő, mivel ez bemutatja, hogy az érintett területen a hallgató nem csak egyszerűen az oktató által bemutatott mintákat követi, hanem a tanult ismeretek elsajátításán túl saját, egyedi gondolatokat is meg tud fogalmazni. Ezáltal az érintett hallgató megoldását ebből a tárgyból akkor is el kell fogadni, hogyha az teljesen másképp működik, mint az oktatói megoldás.

3.3. Visszajelzés típusa

Míg szakmódszertani szempontból fontos lenne a kvalitatív visszajelzés⁴ biztosítása a hallgató részére, az ilyen értékelőrendszerek elsősorban kvantitatív visszajelzés⁵ adására alkalmasak (4.1.3. szakasz).

Egy ilyen rendszer kiválóan alkalmas lehet szummatív visszajelzés adására, vagyis a tanulási folyamat lezárásaként írt zárthelyi dolgozatok értékelésének előkészítésére.

Jellemzően az oktatók az órai feladatok megoldása során biztosítanak formatív visszajelzést, és probléma esetén tudják segíteni a hallgatót.

Egy ilyen jellegű rendszer kiválóan alkalmazható fejlesztői funkciójú értékelésre, mivel a hallgató a saját teljesítményére vonatkozóan kaphat visszajelzést a tanulási szakasz során, és igazodik a hallgató egyéni tanulási folyamatához.

³*Data Manipulation Language*: adatmódosító lekérdezések, például: INSERT, UPDATE, DELETE

⁴Kvalitatív visszajelzés során a hallgató jellemzően szöveges visszajelzést kap arra vonatkozóan, hogy mit csinált jól, mit kellett volna másképp csinálnia, miképpen javíthatta volna megoldását.

⁵Kvantitatív visszajelzés során a hallgatói teljesítményt egy számadattal (például pontszám, százalékos eredmény, osztályzat) szokás értékelni.

Egy ilyen jellegű rendszer alkalmas lehet diagnosztikus értékelésre is, amivel felmérhető a hallgatók előzetes tudása. Ezzel ki lehet alakítani különböző csoportokat (például kezdő, középhaladó, haladó), és azt is vizsgálni, hogy melyek azok a témakörök, amikkel a félév során célszerű többlet foglalkozni. Illetve tanulási problémák esetén különböző témakörökből több feladat feladása alapján a hallgatónak javaslatot lehet tenni, hogy milyen témakört tekintsen át jobban. Több feladat áttekintésével láthatóvá válnak a mintázatok.

A rendszertől egyelőre mindössze annyi az elvárás, hogy kvantitatív visszajelzést adjon. Ezt a zárthelyi dolgozatok szummatív értékelésére szeretnénk használni. És a hallgatóknak biztosítani kívánunk fejlesztői funkciójú visszajelzést olyan módon, hogy a gyakorlaton megoldandó feladatok egy részét ellenőriztethetik a rendszerrel.

Egy ilyen értékelővel kvalitatív és formatív visszajelzés adását további kutatási iránynak jelöljük ki, ahogy a diagnosztikai funkciójú értékelést is.

3.4. Iparban alkalmazható nyelvek használata

Az Információsrendszerek Tanszék elvárása, hogy olyan adatbáziskezelő-rendszer használatával történjen az oktatás, ami a való életben is jól alkalmazható, valamint az is, hogy SQL-ben írt procedurális eljárásokat is lehessen a módszerrel értékelni.

Procedurális SQL esetén lehetőség van az adatbázis-kezelőn futtatni imperatív programozási paradigmának megfelelő kódokat. Vagyis van lehetőség benne szekvenciákat, ciklusokat, értékadásokat, függvényhívásokat is elhelyezni. Oracle Database esetén ezt PL/SQL-nek nevezik, ugyanakkor más adatbázis-kezelőkön (például MySQL, PostgreSQL) is van lehetőség ilyen kódokat írni. Ezek lehetnek eljárások és lehetnek függvények is.

3.5. Nem tökéletes megoldások értékelése

Elvárás, hogy az alap gondolatmenete a hallgatónak meglegyen, a szükséges nyelvi elemeket használni tudja, és a számára hozzáférhető esetben helyes eredményt adjon beküldött megoldása. Ugyanakkor kisebb hibákat véthet a megoldásában, és ebben az esetben is szükséges a megoldását értékelni, akár részpontszámmal. Nem cél, hogy a hallgató az összes szélsőséges esetet megfelelően kezelje a zárthelyi dolgozat megírása során.

Ez az elvárás hasonló, mint amit a hazai középiskolai versenyeken alkalmaznak, amelyet a Bíró és a Mester rendszer működéséről szóló művükben (Horváth és mtsai., 2017)

is rögzítettek. Kivételt a regionális és nemzetközi diákolimpiák jelentik, melyek során a cél, hogy csak az érintett (rész)feladatot tökéletesen megoldó beadások szerezhessenek pontot. Ezt az elvárást az elmúlt években a hazai diákolimpiai válogatón is bevezették.

3.6. Oktatói felügyelet

Fejlesztői funkciójú visszajelzés esetén, ha a hallgató nehézségbe ütközik, akkor segítséget kérhet az oktatótól.

Szummatív értékelés esetén nem elvárás, hogy a rendszer teljes mértékben automata módon adjon az osztályzatokat, hanem mindössze annyit várunk el tőle, hogy egy megalapozott javaslatot adjon a hallgató eredményének értékelésére.

Így a hallgatók számára nem éri meg olyan megoldásokat készíteni, ami alapvetően nem a feladat megoldására irányul, mivel az oktató által is ellenőrizve lesz a megoldása. Ezáltal a zárthelyi dolgozat szűk időkeretében jobban megéri a hallgatónak a feladatot ténylegesen megpróbálni megoldani, és nem a tesztelőt megkerülni.

Az oktatói munkát egy előzetes vélemény elkészítése is jelentős mértékben tudja gyorsítani.

4. fejezet

Értékelési módszerek

4.1. Szakirodalmi áttekintés

Programok helyességének vizsgálatával sok korábbi kutatás foglalkozott. A helyesség vizsgálatát sok esetben egy szoftver új verziójának kiadása előtt automatikusan elvégzik a CI/CD pipeline-okban¹, ezzel biztosítva, hogy a kód várhatóan megfelelően fog futni az éles környezetben is. (Kumar és Mishra, 2016; Wang és mtsai., 2022) munkája alapján az automata tesztelés bevezetése jellemzően javítja a projektek kódjának minőségét, csökkenti a költségeket és a forgalomba hozatalhoz szükséges időt.

Az adatbáziskezelés-feladatok automatikus értékelésekor sok esetben hasonló jellegű problémákat kell megoldani, mint általános célú programozási nyelvek esetén, így az ezzel kapcsolatos szakirodalmat is áttekintjük.

Az automata tesztelést nagy sikerrel vezették be több esetben a programozásoktatásban. Ezek kiterjedtek mind általános célú programozási nyelveken írt megoldások automatizált értékelésére, mind adatbáziskezelés-feladatok automatikus értékelésére. Ezekben a rendszerekben jellemzően a hallgató megtekinthette a számára kitűzött feladatot, saját számítógépén megoldhatta, majd értékelés céljából a tesztelőnek átadhatta. Szakmódszertani szempontok mentén kaphatnak bizonyos megkötéseket (ilyen lehet például a beküldések számának korlátozása). A kapott eredmény alkalmazható akár arra is, hogy a hallgató visszajelzést kapjon, mennyire felkészült az érintett témában, szükséges-e

¹A CI (Continuous Integration) a folyamatos integrációt, a CD (Continuous Delivery / Continuous Deployment) a folyamatos szállítást vagy a folyamatos telepítést jelenti. Ezek célja, hogy a fejlesztők közötti munka összehangolására használt központi verziókövető rendszerbe (például Github, Gitlab) felhelyezett kódon az elkészült tesztek le legyenek futtatva (CI), és akár automatikusan új verziók legyenek kiadva, illetve a teszt rendszerbe vagy az éles környezetbe legyenek telepítve (CD). (<https://qualysoft.com/hu/blog/cicd-bemutatjuk-az-automatizalt-integracio-szallitas-es-telepites-elonyeit>, letöltés dátuma: 2024. május 14.)

még tanulnia az érintett témakört, de akár egy számonkérés szummatív értékelését is végezheti egy ilyen rendszer. (Haldeman, 2021; Horváth és mtsai., 2017; Kádek és mtsai., 2017; Király és mtsai., 2022; Korom, é. n.; Pánovics és mtsai., 2022; Shah és Pareek, 2022)

Alább a különböző, korábban alkalmazott módszereket tekintjük át. Bemutatjuk, hogy milyen előnyeik és hátrányaik vannak az általuk alkalmazott módszereknek, valamint vizsgáljuk, hogy szakmódszertani elvárásainknak megfelelnek-e.

4.1.1. Statikus kódanalízis (white-box tesztelési módszer)

Statikus kódanalízis során a program kódját annak lefuttatása nélkül ellenőrzi. Ezt általános célú programozási nyelvek esetén egyes gyakori hibák megkeresésére és a kód transzformációjára szokás alkalmazni. De nem alkalmas arra, hogy tetszőleges program helyességét eldöntse, mivel ezek jellemzően Turing-teljes nyelvek. Ezáltal általános esetben nem dönthető el, hogy két program ekvivalens-e a megállási probléma miatt egy Turing-gépen. Jelenleg nincs ismert, a Turing-gépnél erősebb számításelméleti modell, és amennyiben igaz a Church–Turing-tézis, akkor nem is lesz a jövőben sem eldönthető az általános eset.

Ugyanakkor nem minden SQL dialektus Turing-teljes, és sok esetben mindössze DQL lekérdezéseket támogattak korábbi értékelők. (Chandra és mtsai., 2016; Sadiq és mtsai., 2004; Shah és Pareek, 2022)

(Shah és Pareek, 2022) munkájukban egy egyedi SQL nyelvet készítettek, amely egyszerűbbé teszi, hogy relációs algebra használatával vizsgálják a helyes megoldás és a hallgató megoldása közti ekvivalenciát, és így képesek akár kvalitatív visszajelzésre is. (Chandra és mtsai., 2016) szintén hasonló elvekkel voltak képesek automata értékelést végezni. (Sadiq és mtsai., 2004) heurisztikus algoritmusokat alkalmaztak arra, hogy megállapítsák, hogy a helyes megoldás és a beküldött megoldás ekvivalens-e.

(Korom, é. n.) munkájában az oktató több különböző, helyes megoldást kellett hogy készítsen. Ezt követően a rendszer a hallgatói kódot és az oktatói kódokat tokenizálta, fát készített belőlük, majd a hallgatói kód és az oktatói kódok között kiszámolta a szerkesztési távolságot, és ez alapján adott meg egy pontszámot. (Minél kisebb volt a távolság, annál jobb volt a hallgató megoldása.)

Ezek a megoldások hasznosak, és DQL lekérdezések értékelésére alkalmasak, és az SQL nyelv többi elemének tesztelésére is adaptálhatóak. Ugyanakkor (Shah és Pareek,

2022) kutatásával ellentétben saját egyedi SQL nyelvet nem alkalmazhatunk, és procedurális SQL-ben írt programok értékelésére a többi bemutatott statikus kódanalízisen alapuló megoldás sem alkalmas.

Az Adatbázisok I tárgy során a félév második felében jelentős részben az Oracle Database által támogatott PL/SQL nyelven írt kódok írását tanulják a hallgatók, és a második zárthelyi dolgozat is jelentős részben ilyen nyelven írt kódok írásából áll. Az elérhető state-of-the-art megoldások nem teszik lehetővé, hogy megbízható visszajelzést biztosítsunk statikus kódanalízis segítségével. Míg hasznos lenne white-box alapú tesztelés biztosítása SQL utasítások esetén, problémát jelenthet, hogy a félév közepén a visszajelzés minősége leromlana a PL/SQL programokkal kapcsolatos feladatoknál, így a statikus kódanalízis alkalmazását elvetjük, és jövőbeli kutatási iránynak jelöljük ki.

4.1.2. Dinamikus ellenőrzés (black-box tesztelési stratégia)

A black-box tesztelési stratégia egy gyakran alkalmazott módszer szoftverek helyességének vizsgálatakor. Ennek során a tesztelendő kódot valamilyen jól meghatározott környezetben és helyzetben futtatják, és vizsgálják, hogy az adott helyzetre a kód az elvárt módon reagál-e.

Ennek a tesztelési módszernek előnye, hogyha volt egy korábban problémát okozó helyzet, akkor lehetőség van néhány ezt a helyzetet modellező tesztesetet készíteni, hogy a jövőben ugyanez ne okozhasson problémát éles környezetben. Ugyanakkor ennek a megoldásnak hátránya, hogy olyan kódok is átjuthatnak a tesztelésen, amelyek nem működnek megfelelően. De a tapasztalat szerint az emberek által végzett tesztelés, a kódminőség ellenőrzése és az automatizált tesztelési megoldások egymást jól kiegészítik. (Kumar és Mishra, 2016; Wang és mtsai., 2022)

A hazai és nemzetközi középiskolai programozási versenyeken (például Nemes Tihamér NITV², OKTV³, Nemzetközi Informatikai Diákolimpia⁴, Közép-Európai Informatikai Diákolimpia⁵, olimpiai válogatóversenyek⁶) valamint egyetemi versenyeken (például ICPC⁷) a programozási feladatokat black-box tesztelési stratégiával értékelik. Ennek előnye, hogy emberi beavatkozás nélkül, azonnali visszajelzést kapnak a ver-

²<http://nemes.inf.elte.hu/>, letöltés dátuma: 2024. május 14.

³https://www.oktatas.hu/koznevels/tanulmanyi_versenyek/oktv_nyito, letöltés dátuma: 2024. május 14.

⁴<https://ioinformatics.org/>, letöltés dátuma: 2024. május 14.

⁵<http://ceoi.inf.elte.hu/>, letöltés dátuma: 2024. május 14.

⁶http://tehetseg.inf.elte.hu/valogatok/valogatok_main.html, letöltés dátuma: 2024. május 14.

⁷<https://icpc.global/>, letöltés dátuma: 2024. május 14.

senyzők beküldött megoldásukra. Így teljesen egzakt módon történhet a tanulói munka értékelése. (Horváth és mtsai., 2017)

Ezen túl a versenyekre való felkészüléskor is használhatóak ezeknek a rendszereknek a nyilvános változatai (például Mester⁸, ProgCont⁹), valamint bevezették őket az egyetemi oktatásban is (például Bíró¹⁰, TMS¹¹). (Horváth és mtsai., 2017; Pánovics és mtsai., 2022; „Task Management System”, é. n.)

Ezek a tesztelők rendre azt az elvet követik, hogy a beérkező kódot szükség esetén lefordítják, adott tesztesetekre lefuttatják, majd ellenőrzik, hogy helyes eredményt ad-e. (Horváth és mtsai., 2017; Mareš, 2021; Pánovics és mtsai., 2022) A beküldött kóddal a kommunikáció történhet az alapértelmezett be-, illetve kimeneten, vagy például sok nemzetközi verseny esetén a versenyző megfelelő függvényeket definiál az előírt programozási nyelven, és a kommunikációt a szervezők kódja végzi a be-, illetve kimeneten keresztül. (Mareš, 2021)

Hasonló elvet követtek (Kádek és mtsai., 2017; Király és mtsai., 2022) is adatbáziskezelés-feladatok automatikus értékelésekor.

DQL lekérdezések esetén az ellenőrzéskor kialakítottak valamilyen helyzetet (kialakították az adatbázis sémát, és feltöltötték adatokkal), lefuttatták a hallgatói kódot, amely eredményét összehasonlították az oktató kódjának eredményével.

DML és DDL¹² utasításokat úgy ellenőriztek (Kádek és mtsai., 2017), hogy kialakítottak valamilyen helyzetet, lefuttatták a hallgató által megadott lekérdezést, majd összehasonlították az adatbázis állapotát azzal, ami az oktatói megoldás után jött létre. Ehhez a megfelelő DQL lekérdezést az oktató adhatta meg.

(Király és mtsai., 2022) nem írták le részletesen a DML és DDL ellenőrzésének menetét.

(Kádek és mtsai., 2017) leírták, hogy volt igény arra, hogy egy megoldást több tesztesettel értékeljenek. (Király és mtsai., 2022) nem tettek említést arról, hogy több tesztesettel is végeznének tesztelést.

(Horcsin és Abonyi-Tóth, 2023) munkánkban egy adatbázis-kezelőtől függetlenül alkalmazható módszerről mi magunk is beszámoltunk, mely során a 4.2. szakaszban bemutatott módszer korábbi változatát javasoltuk, amelyet jelen dolgozatban kiegészítünk.

⁸<http://mester.inf.elte.hu/>, letöltés dátuma: 2024. május 14.

⁹<https://progcont.hu/>, letöltés dátuma: 2024. május 14.

¹⁰<http://biro.inf.elte.hu/>, letöltés dátuma: 2024. május 14.

¹¹<https://tms.inf.elte.hu/>, letöltés dátuma: 2024. május 14.

¹²*Data Definition Language*: adatdefiniáló lekérdezések, például: CREATE TABLE, ALTER TABLE, DROP TABLE

Biztonság

Ezek az értékelők ismeretlen eredetű kódot futtatnak szerveroldalon, így körültekintéssel kell eljárni tervezésükkor, implementációjukkor. Általános célú nyelvek értékelői esetén az, hogy a hallgatói kódot miképpen lehetséges biztonságosan futtatni, az egy részletesen tárgyalt kérdéskör, és a Linux kernel által biztosított state-of-the-art biztonsági megoldásokat már megfelelő módon alkalmazzák. (Mareš, 2021)

A TMS rendszer az értékelés során a beküldött kódokat Docker konténerben futtatja. („Task Management System”, é. n.) Sok automata értékelő a (Mareš, 2021) által bemutatott IOI isolate programot használja sandboxolásra.

(Kádek és mtsai., 2017) részletesen nem mutatták be az alkalmazott biztonsági intézkedéseket (például adatbázis-kezelőn belül alkalmazott jogosultságkezelést). Ennek vélhetően oka, hogy több különböző adatbázis-kezelőt is támogatni kívántak implementációjukban JDBC-kapcsolatokon keresztül, így ezek a beállítások az egyes adatbázis-kezelőkre változóak, vagyis implementációfüggőek. A JDBC a Java Database Connectivity rövidítése, amely lehetővé teszi, hogy a lekérdezéseket és azok eredményét egységes módon kérhessék az adatbázis-kezelő típusától függetlenül. Más nyelvekben is sok esetben van hasonló megoldás.

(Kádek és mtsai., 2017) DML lekérdezések ellenőrzése esetén tranzakcióban hajtották végre a lekérdezéseket. A tranzakciók célja az, hogy az adatbázison futó lekérdezések egy sorozata vagy sikeresen végrehajtsódjon, vagy egyáltalán ne okozzon változást az adatbázisban (atomicitás). Ennek során biztosított, hogy az érintett adatokat más folyamatok nem tudják módosítani (izoláció).

Például egy pénzügyi adatokat tároló rendszerben utalások adminisztrálását mindenképpen célszerű tranzakciókban végezni, mert ellenőrizni kell, hogy a küldőnek van-e elegendő fedezete, ezután le kell vonni tőle az összeget, és növelni kell a címzett egyenlegét.

Ha nem tranzakcióban végeznék az utalást (vagyis nem lenne biztosított az atomicitás), akkor ha rendkívüli esemény történne (például ha természeti csapás nyomán megszűnne a rendszer áramellátása, vagy ha az adatbáziskezelő-motor szoftveres hiba miatt váratlan módon leállna), akkor elképzelhető lenne, hogy a levonás megtörténne, de a fogadó egyenlegének növelése nem.

Ha nem tranzakcióban végeznék (vagyis nem lenne biztosított az izoláció), akkor elképzelhető lenne, hogy két – egyidőben indított – utalás párhuzamos feldolgozása során

mindkettő folyamat megállapítaná, hogy van elég fedezet, és mindkettő levonná az összeget, és mindkettő adna a fogadónak pénzt. De a küldőnek lehet, hogy nem lett volna fedezete mindkettő utalásra, így negatív lesz az egyenlege.

A tranzakciók jellemzően egy megfelelő utasítás kiadásával (például: `START TRANSACTION`) kezdeményezhetőek, és a módosítások adatbázisba írása jellemzően a `COMMIT` utasítással kérhető. Ha a tranzakciót mégsem kívánja a felhasználó az adatbázisba írni, akkor a `ROLLBACK` utasítással állítható helyre az adatbázis kiinduló állapota. A tranzakciók során történő változások adatbázisba írásukig csak az aktuális adatbázissal fennálló kapcsolatban láthatóak.

(Kádek és mtsai., 2017) DML lekérdezések ellenőrzésekor elindítottak egy tranzakciót, elvégezték az ellenőrzéseket, majd visszaállították az adatbázis eredeti állapotát.

Ugyanakkor vannak olyan utasítások, amelyek ún. implicit-committel járnak, vagyis az érintett utasítás hatására az aktuális tranzakcióban történő módosítások az adatbázisba vannak írva. Ezek jellemzően DDL utasítások¹³. Hogy ezek esetén miképpen állították helyre az adatbázis állapotát, azt (Kádek és mtsai., 2017) nem mutatták be.

(Király és mtsai., 2022) leírták, hogy a DQL utasításokat olyan felhasználókkal hajtották végre, akik kizárólag erre rendelkeztek jogosultsággal. A DML és DDL utasításokat ideiglenes táblákon hajtották végre. Az ideiglenes táblák csak az aktuális adatbázissal fennálló kapcsolatban láthatóak, és a kapcsolat bontását követően meg is szűnnek. Az ideiglenes táblák használatának előnye, hogy létrehozható olyan felhasználó, ami véletlenül sem tud olyan írási műveleteket végrehajtani, ami más felhasználók számára látható.

(Kádek és mtsai., 2017; Király és mtsai., 2022) megoldásai lehetővé teszik, hogy párhuzamosan több értékelést futtassanak egyazon adatbázison.

Adatbáziskezelő-rendszerek tervezése során fontos feladat a hozzáférés-kezelés. Ugyanakkor jellemzően lehetősége van egy felhasználónak még alacsony szintű hozzáférés mellett is, hogy olyan erőforrásokat lefoglaljon, amik üzemzavart okozhatnak.¹⁴ Ez egy olyan területe a kiberbiztonságnak, amely nincs részletesen tárgyalva a szakirodalomban – ellentétben a fentebb is leírt, Linuxon történő ismeretlen kódok biztonságos futtatásával (Mareš, 2021) –. Így (Kádek és mtsai., 2017; Király és mtsai., 2022) biztonsági intézkedései ugyan elegendőek lehetnek, de az alkalmazott intézkedések erőteljesen

¹³InnoDB esetén – amely az egyik legelterjedtebb adatbázis-kezelő motor (MySQL és MariaDB mögött is sokszor ez a motor üzemel) – a `CREATE TABLE` és az `ALTER TABLE` utasítások rögtön végrehajtnak, amelyeket nem lehet `ROLLBACK` paranccsal visszaállítani (<https://dev.mysql.com/doc/refman/8.0/en/implicit-commit.html>, letöltés dátuma: 2024. május 3.)

¹⁴Például: <https://stackoverflow.com/questions/18537100/executing-untrusted-sql-with-select-only-user>, letöltés dátuma: 2024. május 3.

függnek az adatbázis-kezelőtől. (Horcsin és Abonyi-Tóth, 2023) művünkben javasoltuk, hogy célszerű magának az adatbázis-kezelőnek a folyamatát konténerizációs megoldásokkal (vagyis (Mareš, 2021) által bemutatott módon Linux kernel namespace-ekkel) vagy egy teljes virtuális gép segítségével (hasonlóan az (Anjali és mtsai., 2020) kutatásában bemutatott módszerhez) korlátozni.

4.1.3. Visszajelzések típusai

A bemutatott rendszerek sok esetben kvantitív visszajelzést, vagyis pontszámot adnak, és tesztetenként tájékoztatják a felhasználót, hogy mi történt programjukkal (például helyes volt az eredmény, helytelen volt az eredmény, túllépte az időlimitet vagy a memórialimitet, vagy futási hiba történt). Ugyanakkor gyakori kutatás tárgya, hogy miképpen lehetne automatikus módon kvalitatív visszajelzést is biztosítani.

(Shah és Pareek, 2022) bemutattak módszert SQL lekérdezésekre adott kvalitatív visszajelzés adására, ugyanakkor mivel Turing-teljes kódot (például procedurális SQL-ben írt kódot) nem képes értékelni, így ezzel kapcsolatban elsősorban az általános célú nyelvekkel kapcsolatos kutatásokra lehet támaszkodni.

(Pánovics és mtsai., 2022) a teszteseteket különböző annotációkkal látták el, amik bemutatták a hallgatónak, hogy milyen jellegű tesztet jelentett számára problémát. Ugyanakkor ehhez egy teljesen egyedi platformot kellett volna készíteni, amelyen keresztül a hallgatókkal tartottuk volna a kapcsolatot, amely a 6.4. szakaszban tárgyaltak alapján sajnos nem volt lehetséges. Így ezt nem valósítottuk meg, de további kutatási irányként ezt a lehetőséget mindenképpen célszerű lenne vizsgálni.

(Koyya és mtsai., 2013) részletesen vizsgálták, hogy miképpen tudnának szöveges visszajelzést biztosítani a beküldött megoldásokra, valamint adtak automatikus korrekciókat is. Ugyanakkor a bemutatott megoldás jelentős mennyiségű fejlesztői munkával járt, és a visszajelzés az esetek felében nem segítette a hallgatók munkáját, a javítások pedig bizonyos feladatoknál kevesebb, mint a hallgatói munkák felénél nyújtott valós segítséget.

A jelenleg elérhető state-of-the-art megoldások nem teszik lehetővé, hogy megbízható kvalitatív visszajelzést biztosítsunk, amelyet a hallgatók is fel tudnak használni. Így elsődlegesen kvantitív értékelési megoldás fejlesztésével foglalkozunk, amelyet a jövőben célszerű lesz kvalitatív visszajelzéssel is kiegészíteni, melyet jövőbeli kutatási irányként jelölünk ki.

4.1.4. Beválásvizsgálatok

Saját tapasztalatunk szerint is, és (Skalka és mtsai., 2019) szerint is sok publikáció készült arról, hogy bevezettek egy-egy értékelőrendszert, de megalapozott beválásvizsgálatot nagyon ritkán publikáltak.

(Sadiq és mtsai., 2004) azt a következtetést vonták le, hogy a hallgatói teljesítmény javult az adatbáziskezelés-feladatok értékelésére bevezetett automata tesztelő hatására.

(Haldeman, 2021) bemutatta, hogy a hallgatók részben fel tudták használni a kvalitatív visszajelzéseket programjuk javítására. Ugyanakkor nem tértek ki, hogy ez segítette-e a hallgatókat a tananyag elsajátításában, vagy mindössze az ellenőrzés során alkalmazott metrika esetén voltak képesek jobb eredményt elérni a hallgatók a kapott visszajelzések segítségével.

(Skalka és mtsai., 2019) vizsgálták, hogy az automata értékelés a hallgatói teljesítményt milyen módon befolyásolja. Megállapították, hogy a hallgatói teljesítményt nem rontotta az, hogy volt fejlesztői funkciójú automata értékelés. Ugyanakkor nem tudtak kimutatni szignifikáns különbséget olyan hallgatói csoportok között, akik rendelkeztek hozzáféréssel egy automata értékelőrendszerhez azokhoz képest, akik nem rendelkeztek hozzáféréssel. Viszont volt összefüggés aközött, hogy valaki használta-e az értékelőt gyakorlásra és aközött, hogy milyen eredményt ért el a számonkérésen.

(Skalka és mtsai., 2019) vizsgálata alapján egy automata értékelőrendszer tesztelésekor jelentős hátrányba nem kerülnek azok a csoportok, amelyeknek nincs hozzáférése a tanulástámogató rendszerhez. Így egy automata tesztelő pilot programban nyugodtan tesztelhető, jelentős hátrányba a programban nem részt vevők nem kerülnek.

(Horváth és mtsai., 2017; Király és mtsai., 2022; Pánovics és mtsai., 2022) megoldásai elsősorban otthoni vagy szakköri gyakorlásra lehet használni. (Skalka és mtsai., 2019) korrelációt mutatottak be a gyakorlás mennyisége és az elért eredmények között, így ezek valóban alkalmasak lehetnek a gyakorlás elősegítésére.

Összességében korábbi eredmények azt mutatják, hogy egy automata értékelőrendszer bevezetése senki számára nem jelent hátrányt. Viszont ha valaki ténylegesen használja gyakorlásra, akkor jellemzően jobb eredményt tud elérni olyan társaihoz képest, akik nem használták a tanulástámogató rendszert.

Ezt a Telekommunikációs Hálózatok tárgy automata értékelőjével kapcsolatos empirikus tapasztalatok is mutatták.

4.2. Javasolt értékelési módszerünk

A vonatkozó szakirodalom részletes vizsgálata alapján az ELTE IK-n az Adatbázisok I gyakorlata számára black-box alapú tesztelést javasolunk, mivel alkalmasabb szakmódszertani elvárásaink teljesítésére, mint a white-box tesztelési módszer.

Felvetődött, hogy white-box alapú tesztelési megoldást is alkalmazzunk, amelyet olyan esetekben használnánk, amikor van rá lehetőségünk. Ugyanakkor a tárgy előrehaladásával jellemzően nem válik könnyebbé a tananyag, de a statikus kódanalízis egyre nehezebbé válik (procedurális SQL kódok esetén pedig lehetetlenné válik a pontos értékelés). Így tantárgypedagógiai szempontból különösebb haszna nem lenne a két módszer vegyes használatának, mivel a második zárthelyi dolgozat nagyrészt procedurális SQL kódok írásából áll.

Ugyan elsősorban Oracle adatbázis támogatását tűzzük ki célul, de olyan értékelési módszert akarunk kialakítani, amely egy-két kisebb módosítással tetszőleges adatbázis-kezelőre adaptálható. Ezért az értékelésre absztrakt lépések bevezetését javasoljuk, amelyet (Horcsin és Abonyi-Tóth, 2023) munkánkban is bemutattunk:

- egy adott tábla tartalmának cseréje adott tartalommal;
- egy adott feladatra a hallgató által beadott választó lekérdezés eredményének ellenőrzése;
- egy adott feladatra a hallgató által beadott módosító lekérdezés végrehajtása;
- az adott SQL dialektusban az oktató által írt választó lekérdezés eredményének ellenőrzése;
- az adott SQL dialektusban az oktató által írt módosító lekérdezés végrehajtása;
- adatbázis visszaállítása eredeti állapotára;
- az adatbázis-kezelő kimenetének ellenőrzése¹⁵.

Állapot előállítása

Az adatbázis visszaállítható egy alapállapotba. Ezután be lehet helyezni a megfelelő táblákba a megfelelő adatokat. Szükség esetén további, adatbázis-kezelőre specifikus utasításokat is végre lehet hajtani.

¹⁵Erre Oracle Database PL/SQL függvények és eljárások esetén volt szükség.

Választó lekérdezések ellenőrzése

Amennyiben választó lekérdezést akar tesztelni az oktató, akkor az adott részfeladathoz tartozó táblák tartalmát szükséges cserélni, a tesztelt lekérdezés eredményét ellenőrizni.

Módosító lekérdezések ellenőrzése

Amennyiben módosító lekérdezést akar tesztelni az oktató, akkor az adott részfeladathoz tartozó táblák tartalmát szükséges cserélni, a tesztelt módosító lekérdezés lefuttatni, és ezután a megfelelő táblák tartalmát kell ellenőrizni. Ha az adatbázis sémája módosul, akkor vissza kell állítani az adatbázis eredeti állapotát, majd a következő tesztesethez tartozó betöltéseket elvégezni, és ez után folytatni tovább az értékelést.

Procedurális SQL függvények tesztelése

Lehetőség van a hallgató által meghatározott függvényt elkészítő utasítást lefuttatni az adatbázison. Létre kell hozni az adatbázis megfelelő állapotát, majd a függvényt le kell futtatni, és ellenőrizni a visszatérési értékét.

Ha a függvény módosította is az adatbázist, akkor a módosító lekérdezésekre vonatkozó ellenőrzést is alkalmazni kell.

Procedurális SQL eljárások tesztelése

Hasonlóan a függvényekhez, a hallgatónak meg kell adnia az eljárást elkészítő utasítást, amelyet le kell futtatni. Az eljárást pedig a megfelelő állapot előállítását követően kell lefuttatni. Ezután a módosító lekérdezésekre vonatkozó ellenőrzést kell alkalmazni.

Több teszteset és több részfeladat kitűzése

Az egyes lépéseket szekvenciálisan egymás után végre lehet hajtani. Egy lépéshez hozzá lehet rendelni egy pontszámot, és így adható a hallgatónak akár részpontszám is. Egy részfeladathoz akár több teszteset is tartozhat olyan módon, hogy több különböző helyzetben is le van futtatva és le van ellenőrizve a hallgatói kód.

4.2.1. Biztonság

4.1.2. szakasz alapján egy automata tesztelő akkor futtathat ismeretlen kódot biztonságosan, hogyha az adatbáziskezelés-feladatok automatikus értékelését végző rendszereket is (Anjali és mtsai., 2020; Mareš, 2021) által bemutatott state-of-the-art biztonsági megoldásokkal implementálják.

Vagyis a Linux kernel által biztosított sandboxolási megoldásokkal (vagy rövid ideig létező virtuális gépekkel) kell korlátozni az adatbázis-kezelő és az értékelést végző program operációs-rendszerbeli folyamatának hozzáférését a különböző erőforrásokhoz. És minden egyes értékelés elején egy új adatbázist kell indítani. Ez természetesen jelentősen növeli a rendszer erőforrásigényét, üzemeltetési költségét ahhoz képest, mintha az adatbázis egy-egy értékelésnél hosszabb ideig létezne. Így célszerű lesz a jövőben olyan adatbázis-kezelőket alkalmazni, amelyek képesek gyorsan beindulni.

Mi a felvázolt problémák miatt Docker konténerben futtatjuk az értékelés során az adatbázis-kezelő és a tesztelést végző szoftver folyamatait.

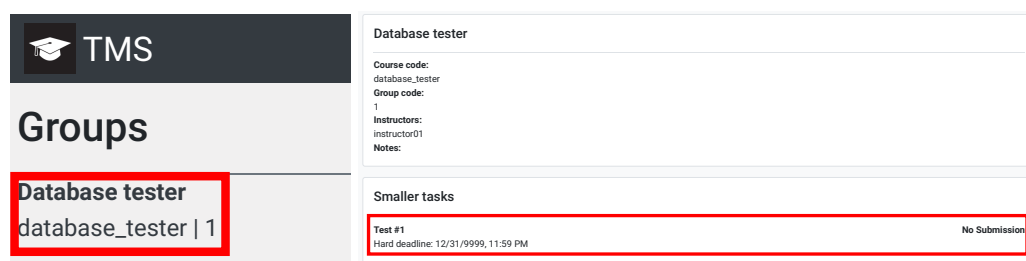
5. fejezet

Felhasználói felületek

A szoftver alapvető célja az, hogy a hallgatók webes felületen beküldve értékelést kapjanak megoldásukra, és az eredményt az oktató is meg tudja tekinteni. A kapcsolattartásra alapvetően a ELTE IK nyílt forráskódú automata tesztelőjét, a TMS rendszert használjuk. A TMS az ELTE IK-n működő webes felületen keresztül elérhető értékelőrendszer, amelybe a hallgatók fel tudnak tölteni programozási feladatokra megoldásokat, és automatikusan visszajelzést kaphatnak beküldött programjukra. („Task Management System”, é. n.)

5.1. Hallgatói felület

A hallgató a TMS rendszerbe belép, a megfelelő csoportra kattintva, a kitűzött feladatot kijelölve (5.1. ábra) meg tudja tekinteni az érintett feladat kezelőfelületét. A 5.2. ábrán tekinthető meg a feladat kezelőfelülete, amely tartalmazza a feladat szövegét, és ahol lehetőség van a megoldás beküldésére.



5.1. ábra. A TMS csoportválasztási, valamint feladatválasztási felülete

A hallgató lehet, hogy olyan feladatot kap, amely során a félév során is alkalmazandó, oktatási célra fejlesztett adatbázis sémát használja, amely tartalmaz minta adatokat

is. Amennyiben nem, akkor a feladat szövege vagy valamely csatolmány tartalmazza az alkalmazandó adatbázis sémát minta adatokkal.

A tanuló az általa választott fejlesztői környezetben megoldja a feladatot. A megoldásként előálló SQL lekérdezéseket a feladatban rögzített formátumú fájlban elhelyezi, a fájlt `submission.sql` néven elmenti. Az így előálló fájlt egy zip archívumba csomagolja, és feltölti a TMS rendszerbe, és beküldi.

A rendszer az értékelést mint oktatói megjegyzés adja meg, amelyről a TMS e-mail értesítést küld, valamint az online felületen is megjelenik az eredmény (5.3. ábra). A pontszám szerinti értékelés tartalmazza az értékelő által adott pontszámot. A részletes megjegyzés pedig tartalmazza az egyes feladatokra az érzékelt karakterkódolást, az egyes feladatokra megoldásként feltöltött SQL kódot, és az értékelő döntését, hogy az érintett részfeladatra helyesnek tekinti-e a kódot. A megjegyzés tartalmazhatja az alább taglalt esetben a zip fájl kicsomagolásához szükséges kódot is, ha az érintett feladathoz szükséges, illetve egy rövid tájékoztató szöveget arról, hogy az értékelőrendszer által adott eredmény csak tájékoztató jellegű.

Test #1

Category:
Smaller tasks
Available:
Soft deadline:
Hard deadline:
12/31/9999, 11:59 PM
Creator:
instructor01

Description:
A feladat megoldását egyetlen `submission.sql` nevű fájlban kell elkészíteni az alábbi formátumnak megfelelően. A fájlnevre valamint a részfeladatok nevére figyelj; mivel ha bármelyik nem megfelelő, az értékelő nem fogja figyelembe venni. Az elkészült megoldást be kell zipelni, majd beküldeni a TMS rendszerben, a `submission.sql` fájl a zip fájl gyökérkönyvtárába kerüljön. A `submission.sql` az alábbi formátumnak megfelelően készüljön el, különben az értékelő nem tudja érzékelni, hogy mi a megoldásod. Egy-egy lekérdezés akár több soros is lehet.

```

task: gyak2_1
sql:
Ennek a szövegnek a helyére kell kerüljön a gyak2_1 feladat megoldása
task: gyak2_2
sql:
Ennek a szövegnek a helyére kell kerüljön a gyak2_2 feladat megoldása

```

Részfeladatok

gyak2_1
Kik azok a dolgozók, akiknek a fizetése nagyobb, mint 2800?

gyak2_2
Kik azok, akiknek a jutaléka nem nagyobb, mint 600?

Upload

Browse

5.2. ábra. A TMS adott feladatnál mutatja a feladat leírását, valamint lehetőséget biztosít a feladat megoldásának beküldésére

Amennyiben az oktatói döntés alapján az egyes hallgatók eltérő feladatsort kapnak, akkor minden hallgatóhoz tartozik a mellékletek között egy fájl, ami tartalmazza a hallgató feladatát titkosított formában. A zip fájl titkosításának feloldására használható jelszót a hallgató megjegyzésként kapja meg (5.4. ábra).

Beadandó értékelés

Új beadandó értékelés került rögzítésre.
Kurzus: Big Data architektúrák és elemző módszerek (csoport: 100)
Státusz: Sikeres tesztelés
Értékelés: 2
Megjegyzés: Az automatikus tesztelés eredménye: Szeretnénk felhívni a figyelmét, hogy az automatikus tesztelés kizárólag az Ön gyakorlásának elősegítése, valamint az oktatói munka egyszerűsítése céljából üzemel. Az automatikus tesztelés eredménye nem feltétlenül jelenti azt, hogy a feladat megoldása helyes vagy teljes. Az automatikus tesztelés eredményétől függetlenül az oktatók el is fogadhatják, de el is utasíthatják megoldását. Ez az értékelés a TMS-be érkezéskor #4 sorszámú feltöltés volt X1HJUJ neptun-kódú, HORCSIN BALINT nevű hallgatótól, mely értékelés 2024-02-28 16:58:51 időpontban készült. Az automata értékelés eredménye: gyak2_1: átmént minden teszten SQL: SELECT dnev FROM dolgozo WHERE fizetes > 2800
 gyak2_2: átmént minden teszten SQL: SELECT dnev FROM dolgozo WHERE jutalek <= 600
 Módosító: Horcsin Bálint

Beadandókezelő rendszer
<https://tms.inf.elte.hu>

Megoldások

Feltöltötte:
Horcsin Bálint (x1hjuj)

Feltöltés ideje:
3/23/2024, 5:43 PM (Europe/Budapest)

Készség:
Sikeres tesztelés

Hitelesített:
Igen

Feltöltések száma:
6

Értékelés:
2

Értékelte:
Horcsin Bálint

Megjegyzések:
Az automatikus tesztelés eredménye:
Szeretnénk felhívni a figyelmét, hogy az automatikus tesztelés kizárólag az Ön gyakorlásának elősegítése, valamint az oktatói munka egyszerűsítése céljából üzemel. Az automatikus tesztelés eredménye nem feltétlenül jelenti azt, hogy a feladat megoldása helyes vagy teljes. Az automatikus tesztelés eredményétől függetlenül az oktatók el is fogadhatják, de el is utasíthatják megoldását. Ez az értékelés a TMS-be érkezéskor #6 sorszámú feltöltés volt X1HJUJ neptun-kódú, HORCSIN BALINT nevű hallgatótól, mely értékelés 2024-03-23 17:44:01 időpontban készült.

Az automata értékelés eredménye:
gyak2_1: átmént minden teszten
SQL: SELECT dnev FROM dolgozo WHERE fizetes > 2800
gyak2_2: átmént minden teszten
SQL: SELECT dnev FROM dolgozo WHERE jutalek <= 600
Git tároló:
A feladat nem verziókezelt

5.3. ábra. A TMS az értékelés eredményéről értesítést küld, és megjelenik az online felületen is

Solution

Name:

Upload time:

Grade:

Status: Rejected

Upload count: 0

Grader: instructor01

Notes: A zárthelyi dolgozat feladatszövegének titkosított állományához használható jelszava: 43732762451784355
Az üzenetet kiküldeni technikai okokból csak értékelés adásával lehet, a zárthelyi dolgozat eredményét nem befolyásolja.

Instructor files

INST01.zip

STUD01.zip

5.4. ábra. A TMS-ben az egyedi fájlok egy-egy titkosított fájlban kerülnek feltöltésre, a hozzá tartozó jelszót a hallgató megjegyzésként kapja meg

5.2. Oktatói felület

Az oktató a hallgatóhoz hasonlóan be tud jelentkezni a TMS rendszerbe, ki tudja választani a megfelelő csoportot és feladatot (5.1. ábra). Az oktató át tudja tekinteni az egyes hallgatók által beküldött megoldásokra az automata értékelő értékelését (5.5. ábra). Az eredmény tartalmazza, hogy az értékelő teszteseteinek mely részfeladatok feleltek meg, illetve a hallgató megoldását is, valamint a pontszám tartalmazza, hogy a tesztelő milyen eredményt javasol a hallgató számára.

Test #1

Edit

Delete

ID:

1

Category:

Smaller tasks

Available:

Soft deadline:

Hard deadline:

12/31/9999, 11:59 PM (Europe/Budapest)

Creator:

instructor01

Password protected:

No

Description:

Solutions

Instructor files

Automatic evaluator

CodeCompass

Solutions

Export

Download Solutions

Sort by

Uploader:

instructor01 (inst01)

Upload time:

4/7/2024, 10:57 PM (Europe/Budapest)

Delay:

Status:

Passed

Verified:

Yes

Upload count:

2

Grade:

3

Grader:

instructor01

Notes:

Az oktató a következő megjegyzést fűzte megoldásához:

A zárthelyi dolgozat feladatszövegének titkosított állományához használható jelszava: 18596474324349057

Az automatikus tesztelés eredménye:

Szeretnénk felhívni a figyelmét, hogy az automatikus tesztelés kizárólag az Ön gyakorlásának elősegítése, valamint az oktatói munka egyszerűsítése céljából üzemel. Az automatikus tesztelés eredménye nem feltétlenül jelenti azt, hogy a feladat megoldása helyes vagy teljes. Az automatikus tesztelés eredményétől függetlenül az oktatók el is fogadhatják, de el is utasíthatják megoldását. Ez az értékelés a TMS-be érkező #2 sorszámú feltöltés volt INST01 neptun-kódú, INSTRUCTOR01 nevű hallgatótól, mely értékelés 2024-04-07 20:57:35 időpontban készült.

Az automata értékelés eredménye:

Érzékelt karakterkódolás: utf-8

task1: átmént a teszteken

Beküldött SQL:

```
SELECT etterem_nev FROM menu WHERE etel_nev = 'zöldséges ragu' INTERSECT SELECT etterem_nev FROM menu WHERE etel_nev = 'palacsinta'
```

task2: nem ment át a teszteken

Beküldött SQL:

```
SELECT etterem_nev, COUNT(*) as db FROM menu NATURAL JOIN recept WHERE alapanyag_nev = 'tojás' GROUP BY etterem_nev
```

task3: átmént a teszteken

Beküldött SQL:

```
SELECT etel_nev FROM recept MINUS SELECT etel_nev FROM recept NATURAL JOIN alapanyag WHERE categoria = 'hús'
```

task4: nem ment át a teszteken

Beküldött SQL:

```
SELECT etterem_nev FROM (SELECT etel_nev FROM recept MINUS SELECT DISTINCT etel_nev FROM recept NATURAL JOIN alapanyag WHERE categoria = 'tejtermék') NATURAL JOIN menu GROUP BY etterem_nev HAVING COUNT(*) >= 2
```

task5: átmént a teszteken

Beküldött SQL:

```
SELECT etel_nev FROM recept GROUP BY etel_nev HAVING COUNT(DISTINCT alapanyag_nev) = (SELECT MAX(db) FROM (SELECT etel_nev, COUNT(DISTINCT alapanyag_nev) db FROM recept GROUP BY etel_nev))
```

IP Addresses:

172.16.0.1

Uploader:

student01 (stud01)

Upload time:

4/7/2024, 10:57 PM (Europe/Budapest)

Delay:

Status:

Passed

Verified:

Yes

Upload count:

2

Grade:

3

Grader:

instructor01

Notes:

Az oktató a következő megjegyzést fűzte megoldásához:

A zárthelyi dolgozat feladatszövegének titkosított állományához használható jelszava: 43732762451784355

Az automatikus tesztelés eredménye:

Szeretnénk felhívni a figyelmét, hogy az automatikus tesztelés kizárólag az Ön gyakorlásának elősegítése, valamint az oktatói munka egyszerűsítése céljából üzemel. Az automatikus tesztelés eredménye nem feltétlenül jelenti azt, hogy a feladat megoldása helyes vagy teljes. Az automatikus tesztelés eredményétől függetlenül az oktatók el is fogadhatják, de el is utasíthatják megoldását. Ez az értékelés a TMS-be érkező #2 sorszámú feltöltés volt STUD01 neptun-kódú, STUDENT01 nevű hallgatótól, mely értékelés 2024-04-07 20:57:58 időpontban készült.

Az automata értékelés eredménye:

Érzékelt karakterkódolás: utf-8

task1: nem ment át a teszteken

Beküldött SQL:

```
SELECT etterem_nev FROM menu WHERE etel_nev = 'zöldséges ragu' INTERSECT SELECT etterem_nev FROM menu WHERE etel_nev = 'palacsinta'
```

task2: átmént a teszteken

Beküldött SQL:

```
SELECT etterem_nev, COUNT(*) as db FROM menu NATURAL JOIN recept WHERE alapanyag_nev = 'tojás' GROUP BY etterem_nev
```

task3: nem ment át a teszteken

Beküldött SQL:

```
SELECT etel_nev FROM recept MINUS SELECT etel_nev FROM recept NATURAL JOIN alapanyag WHERE categoria = 'hús'
```

task4: átmént a teszteken

Beküldött SQL:

```
SELECT etterem_nev FROM (SELECT etel_nev FROM recept MINUS SELECT DISTINCT etel_nev FROM recept NATURAL JOIN alapanyag WHERE categoria = 'tejtermék') NATURAL JOIN menu GROUP BY etterem_nev HAVING COUNT(*) >= 2
```

task5: átmént a teszteken

Beküldött SQL:

```
SELECT etel_nev FROM recept GROUP BY etel_nev HAVING COUNT(DISTINCT alapanyag_nev) = (SELECT MAX(db) FROM (SELECT etel_nev, COUNT(DISTINCT alapanyag_nev) db FROM recept GROUP BY etel_nev))
```

5.5. ábra. Az oktató egyszerre tudja megtekinteni a megoldásait a hallgatóknak

6. fejezet

Hatásvizsgálat

A hatásvizsgálat során a két célcsoport (oktatók és hallgatók) munkájában történt változásokat, valamint a javítás pontosságát vizsgáltuk.

A hatásvizsgálatot az 1-es, 2-es, 4-es és 5-ös sorszámú csoportokban végeztük a 2023/2024-es tanév tavaszi félévében. A csoportok kiválasztásának a szempontja az volt, hogy mindegyik ugyanahhoz az oktatóhoz (Dr. Vörös Péter) tartozzon.

Az 1. és 2. csoportok a Neumann tehetséggondozó programban részt vevő hallgatók, a 4. és 5. csoportok pedig ún. „fixcsoportos” hallgatók voltak¹. Az 1. csoport A-szakirányos, a többi vizsgált csoport B-szakirányos volt.²

A javítás pontosságát az első zárthelyi dolgozat gépes része alapján vizsgáltuk.

A hallgatók véleményét kérdőíven kérdeztük meg. A kérdőíveket az első zárthelyi dolgozatot követő órán töltötték ki papíron. A papíron való kitöltés célja az volt, hogy minél nagyobb legyen a kitöltők száma, ugyanakkor a hallgatók – igény esetén – megőrizhessék anonimitásukat, ezzel növelve a válaszok őszinteségét. A kiadott kérdőívek tartalmát az A. melléklet tartalmazza.

Célunk annak vizsgálata volt, hogy egy ilyen rendszer bevezetése valóban nem jár-e hátránnyal, és milyen előnyökkel jár.

¹Az ELTE IK-n programtervező informatikus nappali alapképzésen tanuló hallgatók számára elérhető a Neumann tehetséggondozó program, amely tagjai részére külön csoportokat indít a Kar. A tárgyfelvétel egyszerűsítése céljából lehetősége van a többi – mintatanterv szerint haladó – hallgatónak is kérni a kötelező tárgyak felvételét a Tanulmányi Hivataltól, akik fixcsoportokban helyezik el az igénylőket.

²Az ELTE IK-n programtervező informatikus nappali alapképzésen három szakirány közül (A, B, C) választhatnak a hallgatók, amelyek közül a modellező (A) szakirányon a matematikai, a szoftvertervező (B) szakirányon a számítástudományi, szoftverfejlesztő (C) szakirányon az informatikai ismeretkör van előtérben. A hallgatók a második félévüket követően választják meg szakirányukat.

6.1. A javítás pontossága

A javítás pontosságát azt követően vizsgáltuk, hogy már az implementációs problémák miatt jelentkező esetleges felhasználói és technikai hibákat orvosoltuk.

Összesen 71 hallgató adott be megoldást, ami feladatsoronként 5-5 részfeladattal számolva 355 értékelt részfeladatot jelent. Ebből 11 részfeladatra nem érkezett megoldás, 336 értékelés eredménye megegyezett a manuálisan adott pontszámmal, 8 esetben kellett eltérni a rendszer által javasolt pontszámtól. Ebből 2 esetben a feladat kitűzőjeként vétettünk hibát; amelyből egyszer a minta megoldás nem volt tökéletes, és így az egyik helyes megoldást nem fogadta el ennek nyomán a tesztelő; egyszer pedig félreérthető volt a feladat (ekkor 1 – 1 ponttal több járt az eredeti értékeléshez képest a hallgatónak). 2 esetben a hallgató bent hagyott néhány felesleges utasítást, amelyektől eltekintettünk, és elfogadtuk a megoldást (ekkor 1 – 1 ponttal több járt az eredeti értékeléshez képest a hallgatónak). 3 esetben kellett csökkenteni a hallgatói pontszámot 0,5 ponttal, mivel relatíve súlyos hibát vétettek egy szűrésben (ez egy típushiba volt). 1 esetben történt olyan, hogy a hallgató megoldása egyáltalán nem oldotta meg a feladatot, hanem egy teljesen véletlenszerűen készített lekérdezés a helyes eredményt adta (ekkor 1 ponttal kevesebb járt az eredeti értékeléshez képest a hallgatónak).

Vagyis az értékelő az esetek 98%-ában helyes eredményt adott, és jellemzően a többi esetben sem tévedett súlyosan, amelyet egy gyors manuális ellenőrzés ne tudott volna orvosolni.

A feladatkitűzéskor ugyanakkor különösen oda kell figyelni, hogy a mintamegoldás helyes legyen, valamint hogy ne legyen félreérthető a feladat. A hallgatóknak hozzá kell szokniuk előzetesen az automata tesztelőhöz, hogy ne hagyjanak bent felesleges utasításokat, oszlopokat, különben az automata értékelő nem fogja tudni értékelni. A jövőben célszerű lehet a nem elfogadható típushibák szűrésére plusz tesztek készíteni.

Célszerű olyan helyzetet előteremteni, hogy ne érje meg a hallgatóknak olyan kódot készíteni, amely csak véletlen megy át a tesztelőn. A 355 értékelt részfeladatból 1 darab olyan volt, amely megpróbálta megkerülni a tesztelőt, ennél sokkal rosszabb arányra számítottunk. Vélhetően a hallgatók meg sem próbálták megkerülni a feladat megoldását, mert tudták, hogy az oktató is át fogja nézni megoldásukat. Továbbá nem érte meg a hallgatóknak megpróbálni megkerülni, mivel a zárthelyi dolgozat során alapból szűk időkeret állt a rendelkezésre, és sokszor nem könnyebb hibás lekérdezést készíteni, amely helyes eredményt ad, mint megírni a helyes megoldást. Ezt a kísérletet meg lehetett volna

akadályozni, ha több tesztet lett volna, de jelen adatok alapján nem tűnik szükségesnek.

Mindenképpen célszerű manuálisan is áttekinteni a hallgatói megoldásokat, hogy valóban a feladatot oldották-e meg. Vagy legalábbis célszerű elhitetni a hallgatókkal, hogy egy ilyen ellenőrzés megtörténik.

A javítás pontossága várakozásainkat jelentősen felülmúlta, ez alapján **a rendszer alkalmas lehet szélesebb körű alkalmazásra.**

6.2. Oktatói munka

A rendszer fejlesztése és üzemeltetése humánerőforrást igényel, ugyanakkor egy ilyen jellegű, jól tervezett rendszer jellemzően jól skálázható.

A 4. és 5. csoport számára a javításkor az eddig szokásos kézi javítással végeztük az ellenőrzést, és az adatbázis-kezelőben kézzel teszteltük, hogy a megoldásként kapott SQL kód valóban lefut-e, helyes eredményt ad-e, és az SQL kód jónak tűnik-e, és jó feladatot oldott-e meg. Az 1. és 2. csoport esetében már kizárólag az értékelő ellenőrizte, hogy fut-e az SQL kód és helyes eredményt ad-e, és manuálisan csak azt ellenőriztük, hogy az SQL kód jónak tűnik-e, és jó feladatot oldott-e meg.

A 37 dolgozat manuális értékelése mintegy 3 órát vett igénybe, a 35 dolgozat félautomata ellenőrzése pedig kevesebb, mint 1 órát az Adatbázisok I gyakorlat egy kezdő demonstrátorának. De a manuális értékelés eredményében volt többször olyan hiba, amelyre az automata értékeléssel való összefésülése során derült fény. A félautomata ellenőrzés során a legnagyobb problémát az jelentette, hogy különböző felületeken kellett azt megkeresni, hogy a hallgató valóban a számára kitűzött feladatot oldotta-e meg. Ezt az értékelőrendszer egy jobb implementációjával lehetséges lesz javítani.

Az oktatói munkát egyszerűsíti a rendszer, így **az oktatói munka szempontjából előnyös a rendszer szélesebb körű bevezetése.**

6.3. Hallgatói vélemények

A hallgatóknak több különböző kérdést is feltettünk, amely során meg kellett határozniuk, hogy egy adott állítással 1-től 4-ig terjedő skálán milyen mértékben értenek egyet. A kérdőív pontosan bemutatta a válaszlehetőségek jelentését, hasonló adatfelvételi meg-

oldással már jó eséllyel találkoztak az OHV-kérdőívek³ kitöltése során, így pontosan ismerték, hogy az egyes válaszlehetőségek mit jelentenek. A kérdőívvel kapcsolatban nem érkezett visszajelzés, hogy nehézséget jelentett volna az értelmezés.

A kérdőívek tartalma függött attól, hogy A- vagy B-szakirányos volt-e az illető. Előbbin a matematikai, utóbbin a számítástudományi ismeretkör van előtérben, szakirányukat a hallgatók a második félévüket követően választhatják. Voltak közös, és voltak az adott szakiránnyal kapcsolatban feltett specifikus kérdések is.

6.3.1. Alkalmazható próbák

Az A-szakirányra specifikus kérdésekre adott válaszok értékelése során kizárólag nem parametrikus próbák voltak végezhetőek (például Wilcoxon-próba) a kis mintaelemszámra való tekintettel (13 fő).

Azokban az esetekben, ahol B-szakirányosok (49 fő) is megkapták a kérdést, a centrális határeloszlás tételét alkalmazhattuk, mivel kellően nagy a minta, és az egyéb feltételeknek is megfelel. A hallgatóknak nem állt érdekében, hogy ne írjanak igazat⁴. Egymást nem tudták érdemben befolyásolni, mivel adatfelvétel közben nem volt lehetőségük összebeszélni, ezáltal a különböző hallgatói válaszok függetlenek egymástól. És az adatok szórása is véges volt, mivel kvázi biztosan tudták, mit szeretnének bejelölni, mivel kevés opció közül kellett egyet kiválasztaniuk egy lineáris skálán, és nem volt olyan jelenség, amitől radikális eltérések lettek volna a hallgatók között. Lehet, hogy szomszédos válaszlehetőségek között nehezen tudtak dönteni, de akkor is legfeljebb 1-gyel tért volna el a válaszuk.

Így a centrális határeloszlás tétele alapján, ha B-szakirányosok is a kérdezettek voltak, alkalmazhattunk T-próbát attól függetlenül, hogy nem tettük fel az adatok normalitását.

Természetesen, ha B-szakirányosok is szerepeltek a válaszadók között, akkor is alkalmazhatóak voltak nem parametrikus próbák.

Alkalmazható próbák összefoglalása Ha két adathalmaz közül meg kívántuk állapítani, hogy egyik nagyobb-e a másiknál, akkor:

- ha csak A-szakirányosokat kérdeztünk meg, akkor Wilcoxon próbát alkalmazhattunk (ekkor a mediánok viszonyára vonhattunk le következtetést);

³A hallgatók az „Oktatás hallgatói véleményezése” kérdőívekben anonim módon véleményt formálhatnak az oktató munkájáról a félév végén.

⁴Tisztában voltak vele, hogy az anonim kérdőívre adott válaszaik alapján retorzió nem érhette őket, de a félév során még fejlődhetett a rendszer, amely miatt számukra könnyebben használhatóvá válhatott a számonkéréseken.

- ha B-szakirányosokat is megkérdeztünk, akkor alkalmazhattunk egymintás T-próbát is a különbségekre (ekkor az átlagok viszonyára vonhattunk le következtetést)⁵, és Wilcoxon próbát is alkalmazhattunk (ekkor a mediánok viszonyára vonhattunk le következtetést).

Ha egy adathalmaz nagyságrendjéről kívántunk megállapításokat tenni, akkor

- ha csak A-szakirányosokat kérdeztünk meg, akkor előjel próbát alkalmazhattunk (ekkor a medián és egy adott szám viszonyára vonhattunk le következtetést);
- ha B-szakirányosokat is megkérdeztünk, akkor alkalmazhattunk T-próbát is (ekkor az átlag és egy adott szám viszonyára vonhattunk le következtetést)⁶, és előjel próbát is (ekkor a medián és egy adott szám viszonyára vonhattunk le következtetést).

6.3.2. Tárgyak nehézsége

A hallgatókat kérdeztük, hogy mennyire tartották nehéznek a 4. félévben kötelező tárgyakat. A válaszok összesítését a 6.1. ábra tartalmazza. Ez azért volt fontos kérdés, mert vizsgálni szeretnénk volna, hogy milyen mértékben szükséges a hallgatói munka segítése Adatbázisok I gyakorlatból.

A hallgatók 1-től 4-ig értékelték, hogy mennyire értettek egyet azzal, hogy az érintett tárgyat nehéznek tartották.

A H_0 hipotézisnek minden esetben azt választottuk, hogy a két tárgy azonos nehézségű. Szoftvertechnológia tárgy kivételével T-próba esetén az volt a H_1 hipotézisünk, hogy az Adatbázisok I tárgyra az átlag kisebb, mint a másik tárgyra adott átlag, vagyis hogy az Adatbázisok I tárgy könnyebb. Szoftvertechnológia tárgy kivételével Wilcoxon-próba esetén az volt a H_1 hipotézisünk, hogy az Adatbázisok I tárgyra a medián kisebb, mint a másik tárgyra a medián, vagyis hogy az Adatbázisok I tárgy könnyebb. Szoftvertechnológia tárgy esetén a H_1 hipotézis az volt, hogy az Adatbázisok I tárgy és a Szoftvertechnológia tárgy különböző nehézségű (T-próba esetén az átlag különböző, Wilcoxon-próba esetén a két medián különböző).

A B-szakirányosok vizsgálata során, és ahol a teljes adatsokaságot tekintettük, a szokásos $\lambda = 5\%$ értéket választottuk. Amennyiben egy kérdést csak A-szakirányosoknak tettünk fel, akkor a $\lambda = 10\%$ értéket választottuk a kis mintaelemszámra való tekintettel.

⁵Párosítottak voltak az adatok – ugyanattól a hallgatótól származott több válasz is –, és nem volt ismert a szórás és a várható érték.

⁶Mivel nem volt ismert a szórás és a várható érték.

A vizsgált B-szakirányos csoportokat tekintve 95%-os megbízhatóság mellett megállapítható, hogy a hallgatók nehezebbnek tartották a Numerikus módszerek I, Valószínűségszámítás és statisztika, Többváltozós függvénytan, Formális nyelvek és automaták tárgyakat, mint az Adatbázisok I tárgyat. A p-értékek a T-próbák esetén kisebbek voltak a választott $\lambda = 5\%$ értéknél (a p-értékek rendre $3,62 \cdot 10^{-5}$, $6,79 \cdot 10^{-9}$, $1,55 \cdot 10^{-7}$, $4,86 \cdot 10^{-5}$ voltak), ezáltal minden esetben elvetettük a H_0 hipotézist.

A vizsgált A-szakirányos csoportot tekintve 90%-os megbízhatóság mellett megállapítható, hogy a hallgatók nehezebbnek tartották a Numerikus módszerek II, Valószínűségszámítás, Analízis III, Bevezetés a számításelméletbe tárgyakat, mint az Adatbázisok I tárgyat. A p-értékek a Wilcoxon-próbák esetén kisebbek voltak a választott $\lambda = 10\%$ értéknél (a p-értékek rendre $3,22 \cdot 10^{-3}$, $2,28 \cdot 10^{-3}$, $3,23 \cdot 10^{-3}$, $7,86 \cdot 10^{-2}$ voltak), ezáltal minden esetben elvetettük a H_0 hipotézist.

Mind a 4 csoport adatait figyelembe véve 95%-os megbízhatóság mellett megállapítható, hogy a hallgatók nehezebbnek tartották az Operációs rendszerek tárgyat, mint az Adatbázisok I tárgyat. A p-érték a T-próba esetén kisebb volt a választott $\lambda = 5\%$ értéknél (p-érték: $1,13 \cdot 10^{-5}$), ezáltal elvetettük a H_0 hipotézist.

Nem tudtuk elvetni, hogy a Szoftvertechnológia tárgy és az Adatbázisok I tárgy nehezsége egyforma lenne, mivel a p-érték meghaladta a választott $\lambda = 5\%$ értéket. (T-próba esetén a p-érték $3,3 \cdot 10^{-1}$ volt.)

Ahol T-próbát végeztünk, ellenőrzésképpen Wilcoxon-próbát is végeztünk. Ezzel a statisztikai módszerrel is lényegében ugyanazokra az eredményekre jutottunk.

Összességében megállapítható, hogy az Adatbázisok I tárgyat könnyebbnek tartották a Szoftvertechnológián kívül a többi tárgyhöz képest, így a **hallgatói munka segítése ugyan fontos, de nem kritikus**.

6.3.3. Zárthelyi dolgozattal és értékelővel kapcsolatos vélemények

Az értékelő rendszer otthoni használatának hiányát a félév során is láttuk, ezért is tettünk ezt firtató kérdéseket a kérdőívbe. A 6.1. táblázatban helyeztük el a válaszok összesítését.

A szabad szöveges visszajelzések alapján kiderült, hogy a hallgatók nem használták otthoni gyakorlásra a tesztelőt, mivel kizárólag gyakorlaton megoldott feladatok voltak elérhetőek a tesztelőben, és így nem látták értelmét a használatának. Azt sem feltétlen tudták, hogy az óra idején túl is szabad volt használni a tesztelőt.

Magyarázat arra, hogy miért nem használta	Jelölések darabszáma
Nem tudtam, hogy van lehetőség gyakorolni rajta.	35
A tárgyat enélkül is teljesíthetőnek tartottam.	16
Nem volt időm gyakorolni a tárgyra.	9
Más tárgyakba befektetett időt jobban megtérülőnek tartottam.	10
A felület nehezen volt használható.	2
Nem láttam értelmét a használatának.	2

6.1. táblázat. Magyarázatok arra, hogy miért nem használták otthoni gyakorlásra

A 6.2. táblázat tartalmazza a hallgatók által adott visszajelzések darabszámát a további kérdésekre.

A vélemények

Az értékelőrendszerrel kapcsolatos kérdésekre T-próbával vizsgáltuk, hogy az egyes állításokkal milyen mértékben értettek egyet a hallgatók.

A nagy elemszámra való tekintettel a szokásos $\lambda = 5\%$ -ot választottuk. Egy választott s számmal az a H_0 hipotézis, hogy az adott kérdésre az átlag megegyezik s -sel. A H_1 hipotézis pedig az, hogy az s -nél az átlag nagyobb.

Szinte teljes egyetértés volt azzal⁷, hogy:

- igazságosnak tartották a zárthelyi dolgozat értékelését (p-érték: $4,6 * 10^{-4}$);
- nem okozott érdemi nehézséget az egyedi feladatsorok megnyitása (p-érték: $1,6 * 10^{-6}$).

A hallgatók egyetértettek azzal⁸, hogy:

- hasznosnak tartották, hogyha van lehetőség automata értékelővel gyakorolni (p-érték: $6,36 * 10^{-4}$);
- hasznosnak tartották volna, hogyha lett volna mintaZH (p-érték: $1,02 * 10^{-3}$);
- hasznosnak tartották, hogy gyors visszajelzést kaptak a ZH alatt (p-érték: $4,5 * 10^{-11}$);

⁷ $s = 3,5$ választása esetén a p-érték kisebb a választott λ -nál, így H_0 -t elvethetjük. Vagyis az átlag 95%-os valószínűséggel 3,5 fölött lesz, ami szinte teljes egyetértést jelent.

⁸ $s = 3$ választása esetén a p-érték kisebb a választott λ -nál, így H_0 -t elvethetjük. Vagyis az átlag 95%-os valószínűséggel 3 fölött lesz, ami egyetértést jelent. Nem belevéve azokat a szempontokat, amelyekkel teljes mértékben egyetértettek.

- alkalmas volt a zárthelyi dolgozat teljesítményük mérésére (p-érték: $2,72 \cdot 10^{-6}$).

A hallgatók számára jellemzően könnyű volt használni az automata tesztelőt⁹ (p-érték: $3,31 \cdot 10^{-7}$).

A mediánt előjel próbával vizsgálva is hasonló – habár pontatlanabb – eredményeket kaptunk.

Az értékelőrendszer használatával kapcsolatban voltak nehézségek, amelyeket a hallgatók is megemlítettek, erről a 6.4. szakaszban számolunk be.

	1 egyáltalán nem értek egyet	2 inkább nem értek egyet	3 inkább egyetértek	4 teljesen egyetértek	Nem tudok / nem kívánok válaszolni
Hasznosnak tartom, hogy volt lehetőség otthon automata tesztelő segítségével gyakorolni.	2	1	10	22	27
Az egyedi feladatsorok megnyitása nem okozott érdemi nehézséget a zárthelyi dolgozat megírásakor.	1	0	11	48	2
Hasznosnak tartottam volna, hogyha a minta zárthelyi dolgozatot is lehetett volna automatikusan teszteltetni.	3	5	20	32	2
Használtam otthoni gyakorlásra a tesztelőt.	47	6	3	1	5
Hasznosnak tartom, hogy a zárthelyi dolgozat írása során kaptam gyors visszajelzést teljesítményemre.	0	2	20	36	4
Alkalmasnak tartottam a teljesítményem mérésére a zárthelyi dolgozatot.	1	2	28	29	1
Igazságosnak tartom a zárthelyi dolgozat értékelését.	0	1	14	43	3
Könnyű volt használni az automata értékelőt.	5	6	25	25	0

6.2. táblázat. Az értékelőrendszerrel és a zárthelyi dolgozattal kapcsolatos kérdőívre érkező válaszok összesítése

⁹ $s = 2,5$ választása esetén a p-érték kisebb a választott λ -nál, így H_0 -t elvethetjük. Vagyis az átlag 95%-os valószínűséggel 2,5 fölött lesz, ami jellemzően egyetértést jelent. Nem belevéve azokat a szempontokat, amelyekkel már jobban egyetértettek.

6.3.4. A hatásvizsgálat során nem vizsgált csoportok

Ki kell emelnünk, hogy a kérdésfeltétel módja és időpontja alapján a hallgatók elsősorban az érintett tárgyak gyakorlati követelményeit, a gyakorlatok és az előadások nehézségét értékelték, a vizsgákról állást korlátozottan tudtak foglalni. Ugyanakkor azzal, hogy ilyen módon vettük fel az adatokat, az értékelő szempontjából pontos adatokhoz jutottunk, amelyet nem befolyásoltak a vizsgaeredmények, amely problémát jelenthetett volna, hogyha az OHV-kérdőívek eredményeit vizsgáltuk volna.

A kiválasztott csoportok nem reprezentálják az ELTE IK hallgatóit.

A vizsgált csoportok tagjai először vették fel a tárgyat, így akik többször vették fel a tárgyat, azok nem tudtak nyilatkozni a kérdésben. Az Adatbázisok I gyakorlat tárgyat felvevő hallgatók tárgyfelvételi statisztikáit a 6.3. táblázat tartalmazza.

Így a tárgy többszöri felvétele a hallgatók mintegy 5,66%-át érintő jelenség, amely nem elhanyagolható. Elképzelhető, hogy számukra nagy segítséget jelentene, hogyha tudnánk biztosítani további oktatási segédanyagokat. Célszerű lesz a jövőben olyan hatásvizsgálatot tartani, ami azt vizsgálja, hogy azon hallgatók számára az automata tesztelés milyen hatással jár, akik a tárgyat első alkalommal nem tudták sikeresen teljesíteni.

Tárgyfelvevételek száma	1	2	3 vagy több
Hallgatók száma	1833	96	15

6.3. táblázat. IP-18AB1G tárgykódú tárgy felvételi statisztikái

A jövőben célszerű lesz hatásvizsgálatot végezni, amelyben több különböző oktató is részt vesz, valamint ahol reprezentálva vannak a kevésbé jó képességű, valamint a C-szakirányos hallgatók.

6.4. Felmerülő problémák

Megannyi más rendszerhez hasonlóan, ennek is voltak olyan hibái, amelyek a végfelhasználói teszteléskor derültek ki. Az implementációs részletekről nem számolunk be jelen dolgozatban, hanem kizárólag a tantárgypedagógiai szempontból releváns részeket mutatjuk be. További részleteket szakdolgozatom tartalmaz, amelyet jelen dokumentum írásakor még nem véglegesítettem.

Az ELTE IK nyílt forráskódú rendszerén, a TMS-en keresztül küldhettek be megoldásokat a hallgatók, amelyeket egy külső számítógépen lekérdeztünk a TMS API-jából,

és elvégeztük a tesztelést. („Task Management System”, é. n.) Magát az értékelést – a 4.2. szakaszban bemutatottaknak megfelelően – egy Docker konténerben futó adatbázis-kezelőn futtattuk, amelyet minden értékelés végén megsemmisítettünk.

A saját külső kiszolgáló használatának oka az volt, hogy a TMS számára elérhető erőforrások korlátozottak voltak, az Oracle Database túl lassan indult. Az Oracle Database indítását a saját kiszolgálón fel tudtuk gyorsítani, de az általunk alkalmazott megoldást a TMS központi kódjába a fejlesztők nem kívánták integrálni. A rendszer kettéválását a hallgatók korlátozottan érzékelhették csak, de a hibás implementációból eredő problémák esélyét növelte, hogy a TMS-t eredendően nem erre a felhasználási esetre tervezték. A felvetődő igények kielégítésére a TMS-t többször kreatív módon kellett rávennünk, amely csökkentette implementációnk stabilitását, és a hallgatók számára is nehezítette a rendszer használatát.

A különböző problémák egy részét mi magunk észleltük, de egy részét hallgatói oldalról gyűjtöttük a hatásvizsgálat során.

A külső kiszolgáló használatának ötletét a Telekommunikációs hálózatok tárgy értékelőjétől vettük. Abban az esetben jól bevált megoldás volt, hogy a hallgatók Pythonban írt kódjukat feltöltötték a TMS rendszerbe, amelyet egy külső alkalmazás a TMS API-ján keresztül letöltött, majd értékelt. Felvetődött alternatívaként a Canvas LMS használata is¹⁰, de a hallgatóval való kapcsolattartásra alkalmazott platform kiválasztásakor még nem volt garantált, hogy a szükséges jogosultságokkal rendelkezni fogunk a Canvas rendszerben.

Elengedhetetlen volt, hogy a TMS-t vagy a Canvas-t használjuk a hallgatókkal való kapcsolattartásra, mivel a rendszerünknek biztosított hardverrel másképp nem volt megoldható az ELTE Információbiztonsági Szabályzatának való megfelelés.

A 4-es csoportnál felvetődő problémák A 4-es csoport írt elsőként zárthelyi dolgozatot. A számonkérés írása közben hibás automatikus visszajelzést kaptak beküldött megoldásukra. Többen is jelezték a kérdőívben ezzel kapcsolatban nemtetszésüket, mivel időt vett el tőlük a zárthelyi dolgozat írása során.

Ebből azt a következtetést vontuk le, hogy egy hibásan működő automata tesztelő rendszer egy stresszhelyzetben több kárt okoz, mintha nem lenne értékelőrendszer.

A többi csoport zárthelyi dolgozatáig a problémát sikerült orvosolnunk.

¹⁰ A Canvas az ELTE-n szintén elérhető tanulástámogató rendszer.

Fájlformátummal kapcsolatos problémák A TMS rendszer elvárása, hogy a szöveges állomány egy zip fájlba be legyen csomagolva. Ezt több hallgató is nehezményezte a szabadszöveges értékelésben.

A szoftver a 4. és 5. csoport esetében elvárta, hogy a hallgató UTF-8 karakterkódolást alkalmazzon, ellenkező esetben nem tudta értékelni a megoldást. Az 1. és 2. csoport zárthelyi dolgozatakor a rendszer már képes volt automatikusan érzékelni a beküldött fájl karakterkódolását¹¹. Egyes csoportokban több, mint a hallgatók 25%-ának végső megoldása sem UTF-8 kódolással volt mentve. Ez rávilágított arra, hogy egy automata értékelő fejlesztésekor figyelni kell arra, hogy a hallgatók által használt szövegszerkesztők milyen karakterkódolás szerint mentenek alapértelmezés szerint, és fel kell készíteni értelmezésére a rendszert. Javíthatja a helyzetet, hogyha a feladatok úgy vannak kitűzve, hogy ne kelljen az ASCII karaktértáblában nem szereplő karaktereket használni.

Nehézséget jelentett 3 hallgatónak, hogy megfelelő néven mentse el a megoldását, és azt a zip fájl gyökerében helyezze el.

Hiába kaptak a hallgatók többszörös szóbeli utasítást, egy zárthelyi okozta stresszhelyzetben a technikai problémákat nem feltétlen tudták megoldani. Ezért egy automata értékelőrendszernek meg kell próbálnia minél több lehetséges hibás használatot kezelnie, mivel a hallgatótól pusztán ilyen formai hibák miatt korábban sem voltak le pontot, és ezt követően sem tervezzük ezt a tárgy értékelése során. Így ha egy hallgató megoldását a rendszer nem találja meg magától, azzal csak az oktató és hallgató együttműködését nehezítheti egy újabb konfliktus-forrással.

A beküldött fájl formátumára nagyon komoly megkötések voltak, amely a 6.1. forráskódban látható. Ehhez képest javasoljuk, hogy a séma létrehozásának lépései, valamint a feladatsor maga is szerepeljen a hallgató részére odaadott fájlban, jó névvel, megfelelően kommentezve, amit a 6.2. forráskódban mutatunk be. Az új formátum tesztelése jelenleg is zajlik. Reményeink szerint az új formátum hatására a hallgatók számára kevesebb nehézséggel fog járni a megoldás elvárt formátumban történő beadása.

Egyedi feladatsorok kitűzése Mivel az automata értékelővel a TMS rendszeren keresztül tudtak kommunikálni a hallgatók, így megkötötte a TMS rendszer működési módja azt, hogy milyen funkciók voltak számunkra elérhetőek a feladatok kitűzésekor.

A TMS nem támogatja egyedi feladatok kitűzését. Ugyanakkor az akadémiai integritás megőrzése érdekében elvárás volt, hogy a különböző hallgatók ne pontosan ugyanazt

¹¹A <https://pypi.org/project/chardet/> csomag segítségével (letöltés dátuma: 2024. május 8.).

a feladatot oldják meg.

Ezt úgy tudtuk áthidalni, hogy minden hallgató számára az egyedi feladatsorhoz a dokumentumokat előkészítettük, becsomagoltuk mindenki számára egy-egy saját titkosított zip fájlba, és melléeltük a fájlokat a feladathoz, és a rendszerben a titkosítási kulcsot megadtuk a hallgatónak megjegyzésként.

Kizárólag ezt a megoldást találtuk, amellyel garantálni tudtuk, hogy:

- minden hallgató egyedi feladatsort kapjon;
- a hallgató ne tudjon véletlenül rossz feladatsort megoldani;
- a hallgató ne férjen hozzá idő előtt feladatsorához;
- ne kelljen a feladatok eléréséhez a TMS-en kívül egyéb szolgáltatást (például más honlapot) igénybe venni.

Ez nehezítette a hallgatók munkáját, ugyanakkor a hallgatói visszajelzések alapján ez nem jelentett különösebb nehézséget.

6.5. Hatásvizsgálat összefoglalása

A hatásvizsgálat eredményeképpen a következő megállapításokat tudjuk tenni:

- a rendszer alkalmas zárthelyi dolgozatok tesztelésére, ha utólag van oktatói felülvizsgálat;
- az oktatói munkát egyszerűsíti egy ilyen jellegű rendszer;
- a hallgatók hasznosnak tartják, hogyha van automata tesztelő (mind otthoni gyakorlásra, mind a zárthelyi dolgozatok javítására);
- a hallgatók igazságosnak érzik a kialakuló eredményt, amelyet az oktató az értékelőrendszer előzetes véleményének figyelembevételével hoz;
- minél felhasználóbarátabbnak kell lennie a rendszernek, különben lesz hallgató, aki nem fogja tudni megfelelően használni.

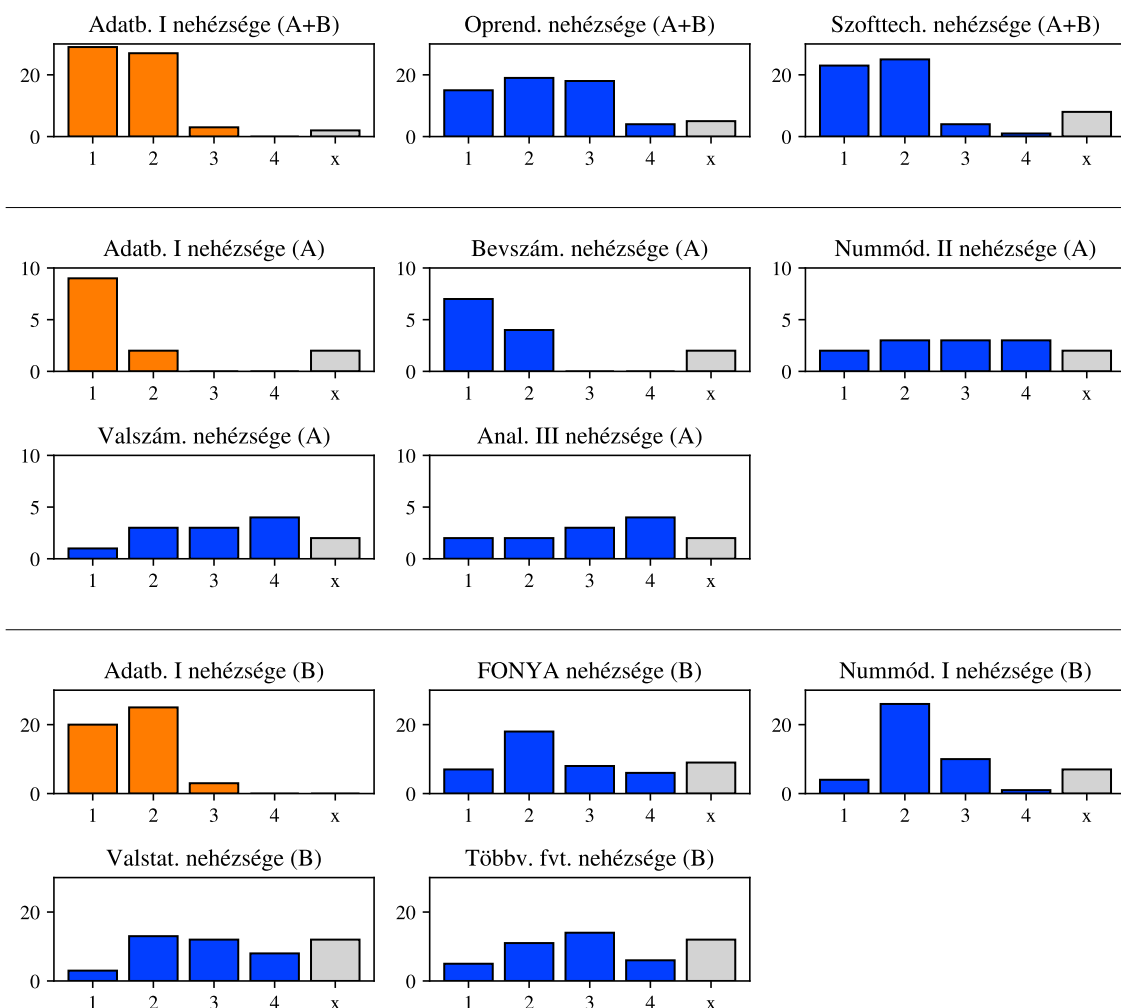
Javasoljuk az általunk javasolt módszer szélesebb körű bevezetését.

```
1 task: task1
2 sql:
3 Ennek a szövegnek a helyére kerüljön task1 megoldása.
4 task: task2
5 sql:
6 Ennek a szövegnek a helyére kerüljön task2 megoldása.
```

6.1. forráskód. A beküldendő fájl formátuma

```
1 CREATE TABLE menu (
2   etterem_nev VARCHAR2(20) NOT NULL,
3   etel_nev VARCHAR2(20) NOT NULL);
4 [...]
5 INSERT INTO menu VALUES ('Vega vagon', 'palacsinta');
6
7 --task: task1
8 --sql:
9 --Feladat: Melyek azok az éttermek ahol zöldséges ragu van, de
   palacsinta nincs az étlapon?
10 Ennek a szövegnek a helyére kerüljön task1 megoldása.
11 --task: task2
12 --sql:
13 --Feladat: Listázza ki a Betyár bisztró ételeinek összesített kaló
   riaértékét (alapanyag_kcal * mennyiség)?
14 Ennek a szövegnek a helyére kerüljön task2 megoldása.
```

6.2. forráskód. A beküldendő fájl új formátuma



6.1. ábra. Az egyes tárgyak hallgatók által tapasztalt nehézsége. Rövidítések: *Adatb.*: Adatbázisok, *Oprend.*: Operációs rendszerek, *Szofttech.*: Szoftvertechnológia, *Bevszám.*: Bevezetés a számításelméletbe, *Nummód.*: Numerikus módszerek, *Valszám.*: Valószínűesszámszámítás, *Anal.*: Analízis, *FONYA*: Formális nyelvek és automaták, *Valstat.*: Valószínűesszámszámítás és statisztika, *Többv. fvt.*: Többváltozós függvénytan. Ha egy hallgató nem kívánt válaszolni, x-ként volt jelölve. Amennyiben a sort üresen hagyta, vagy nem töltötte ki a kérdőívet, akkor nem került bele a statisztikába. Az egyes diagramok címei tartalmazzák zárójelben, hogy milyen szakirányú csoport vagy csoportok adatait tartalmazza.

7. fejezet

Összefoglalás

7.1. Javaslatok

Javasoljuk, hogy megoldható legyen a TMS rendszeren belüli tesztelés. Ehhez szükséges lenne a TMS rendszerben egy-egy tesztelésre szánt idő növelése, vagy pedig az Oracle Database helyett olyan adatbáziskezelő-rendszer alkalmazása, amely képes gyors indulásra tesztkörnyezetben (például MySQL). Utóbbi esetre javasoljuk a programtervező informatikus alapszakos tanterv módosítását, vagyis a PostgreSQL-re vagy MySQL-re való áttérést. Esti képzésen már áttértek PostgreSQL-re az automata értékelőtől függetlenül.

Javasoljuk, hogy legyen lehetőség egyes feladatokat külön-külön szövegmezőben beküldve is feltölteni a TMS rendszerbe. Vagy pedig legyen lehetőség egy-egy fájlt feltölteni, és ne csak zip fájlt. Valamint javasoljuk, hogy a TMS rendszerben közvetlenül lehessen egyedi feladatsorokat kitűzni kreatív megoldások alkalmazása nélkül.

Amennyiben a TMS rendszerben ezek a változások nem megoldhatóak, célszerű lehet az értékelő által használható erőforrások bővítése. Megfontolandó lehet azt is, hogy a hallgatókkal való kapcsolattartásra az egyetemen szintén elérhető Canvas LMS-t használják.

Összességében az általunk alkalmazott módszer kiválóan bevált a hatásvizsgálat során, így **javasoljuk a szélesebb körű bevezetését.**

7.2. Hozzájárulásunk

Áttekintettük, és megfogalmaztuk azokat a szakmódszertani elvárásokat, amelyeknek az ELTE IK-n az Adatbázisok I tárgy oktatásában használt automata értékelőnek meg kell felelnie. Megvizsgáltuk a szakirodalomban fellelhető korábbi megoldásokat, értékeltük módszereiket aszerint, hogy azok az ELTE-n alkalmazhatóak lennének-e. A korábbi megoldások alapján új ötletekkel kiegészítve javasoltunk egy saját módszert, amely jobban illeszkedik az ELTE oktatásmódszertani elvárásainak. Implementáltuk saját megoldásunkat, és végeztünk egy hatásvizsgálatot 72 hallgató bevonásával. Beválasztóvizsgálatunk eredménye alapján javaslatot tettünk megoldásunk bevezetésére, illetve a felmerülő problémák jövőbeli megoldására.

7.3. További kutatási irányok

Célszerű lenne a DQL lekérdezéseken kívül más típusú lekérdezésekkel kapcsolatban is hatásvizsgálatot folytatni. Jelen dokumentum írásakor ez a vizsgálat folyamatban van, ugyanakkor azzal a vizsgálattal kapcsolatban egyelőre következtetéseket levonni nem tudunk. A konferencián viszont várhatóan részeredményeket tudunk már ismertetni.

Hasznos lenne még több hallgató bevonásával beválasztóvizsgálatot tartani. Különösen olyan csoportokban, amelyek nem tehetséggondozók illetve fixcsoportok, valamint C-szakirányon lévő hallgatók tekintetében is célszerű lenne vizsgálatot tartani¹. Kiemelten vizsgálni kellene azon hallgatókra kifejtett hatását, akik nem első alkalommal végzik a tárgyat.

Mivel a rendszer lehetőséget teremthet az Adatbázisok I gyakorlat tárgy esetében is házi feladatok kitűzésére, így vizsgálható, hogy hasonlóan pozitív hatásokat lehet-e elérni, mint a többi folyamatos számonkérésű tárgy esetében.

Célszerű lenne vizsgálni, hogy kvalitatív illetve formatív visszajelzést is adhasson a rendszer, vagyis a hallgató kaphasson szöveges visszajelzést teljesítményére vonatkozóan. Valamint hasznos lenne tágabb feladatgyűjteménnyel biztosítani a fejlesztői célú értékelést. És célszerű lenne diagnosztikai funkciójú értékelést is beépíteni a rendszerbe, amely

¹Az ELTE IK-n a hallgatók számára elérhető a Neumann tehetséggondozó program, amely tagjai részére külön csoportokat indít a Kar. Lehetősége van a többi hallgatónak is kérni a kötelező tárgyak felvételét a Tanulmányi Hivattaltól, akik fixcsoportokban helyezik el az igénylőket. A hallgatók három szakirány közül (A, B, C) választhatnak, az A-szakirányon a matematikai, a B-szakirányon a számítástudományi, C-szakirányon az informatikai ismeretkör van előtérben. A hatásvizsgálatban nem vettek részt C-szakirányos hallgatók és nem fixcsoportozó / tehetséggondozó csoportokhoz tartozó hallgatók sem.

során vizsgálható lenne, hogy milyen jellegű részfeladatokat tud megoldani a hallgató, és az alapján meghatározni a tanulói csoportokat és a tanévben kiemelten kezelt anyagrészeket, vagy javaslatot tenni számára a további átnézendő tananyagrészekre a kialakuló minták alapján.

Hasznos lenne vizsgálni az újabb értékelési eszközök és funkciók beépítésével az értékelő által adott visszajelzés hallgatói teljesítményre gyakorolt hatását.

Köszönetnyilvánítás

Köszönöm témavezetőmnek, Dr. Abonyi-Tóth Andornak, az ELTE-IK Média- és Oktásinformatikai Tanszék egyetemi docensének ötleteit és tanácsait, amelyek nagyban hozzájárultak jelen dolgozat sikeres elkészültéhez.

Köszönöm Dr. Vörös Péter egyetemi adjunktusnak, aki lehetőséget adott, hogy csoportjainál elvégezhessem a bevéásvizsgálatot. Köszönöm továbbá észrevételeit, amely segítette a bevéásvizsgálat sikeres elkészültét.

Köszönöm Dr. Gombos Gergő egyetemi adjunktusnak, hogy bemutatta a Telekommunikációs Hálózatok tárgy automata értékelőrendszerének működését, és hogy biztosított helyet értékelőrendszeremnek.

Köszönettel tartozom továbbá az Eötvös József Collegium Informatika Műhely tagjainak és vezetőjének, Dr. Lócsi Levente adjunktusnak, akik javaslataikkal támogatták jelen dolgozat elkészültét.

„A Kulturális és Innovációs Minisztérium ÚNKP-23-1 kódszámú Új Nemzeti Kiválóság Programjának a Nemzeti Kutatási, Fejlesztési és Innovációs Alapból finanszírozott szakmai támogatásával készült.”



7.1. ábra. ÚNKP, NKFIH és KIM logója

A. függelék

Hallgatóknak kiadott kérdőív

Alább tekinthető meg, milyen kérdéseket adtunk a hallgatóknak. A tördelése más volt az eredeti dokumentumnak, és a csoportok a csoport sorszámát kitöltve kapták meg, és csak a számukra releváns tárgyak szerepeltek.

Csoport száma: **(ide beírtuk a csoport számát)**

Név (nem kötelező): _____

Jelölje az állítással való egyetértésének mértékét! Az állítások elsősorban az Adatbázisok I Gy tárgyhoz használt automata tesztelőjével, valamint az Adatbázisok I Gy tárgy első zárthelyi dolgozatának gépes részével kapcsolatosak.

	1 egyáltalán nem értek egyet	2 inkább nem értek egyet	3 inkább egyetértek	4 teljesen egyetértek	Nem tudok / nem kívánok válaszolni
Hasznosnak tartom, hogy volt lehetőség otthon automata tesztelő segítségével gyakorolni.					
Az egyedi feladatsorok megnyitása nem okozott érdemi nehézséget a zárthelyi dolgozat megírásakor.					
Hasznosnak tartottam volna, hogyha a minta zárthelyi dolgozatot is lehetett volna automatikusan teszteltetni.					
Használtam otthoni gyakorlásra a tesztelőt.					

	1 egyáltalán nem értek egyet	2 inkább nem értek egyet	3 inkább egyetértek	4 teljesen egyetértek	Nem tudok / nem kívánok válaszolni
Hasznosnak tartom, hogy a zárthelyi dolgozat írása során kaptam gyors visszajelzést teljesítményemre.					
Alkalmasnak tartottam a teljesítményem mérésére a zárthelyi dolgozatot.					
Igazságosnak tartom a zárthelyi dolgozat értékelését.					
Könnyű volt használni az automata értékelőt.					
(A-szakirány esetén) Nagyon nehéznek tartom a Numerikus módszerek II tárgyat.					
(B-szakirány esetén) Nagyon nehéznek tartom a Numerikus módszerek I tárgyat.					
(A-szakirány esetén) Nagyon nehéznek tartom a Valószínűségszámítás tárgyat.					
(B-szakirány esetén) Nagyon nehéznek tartom a Valószínűségszámítás és statisztika tárgyat.					
(A-szakirány esetén) Nagyon nehéznek tartom az Analízis III tárgyat.					
(B-szakirány esetén) Nagyon nehéznek tartom a Többváltozós függvénytan tárgyat.					
Nagyon nehéznek tartom az Szoftvertechnológia tárgyat.					
Nagyon nehéznek tartom az Adatbázisok I tárgyat.					
Nagyon nehéznek tartom az Operációs rendszerek tárgyat.					
(A-szakirány esetén) Nagyon nehéznek tartom a Bevezetés a számításelméletbe tárgyat.					
(B-szakirány esetén) Nagyon nehéznek tartom a Formális nyelvek és fordítóprogramok alapjai tárgyat.					

A zárthelyi dolgozaton szerzett pontszáma (nem kötelező): _____

Amennyiben nem használta otthoni gyakorlásra a tesztelőt, akkor annak mi volt az oka?

- ☐ Nem tudtam, hogy van lehetőség gyakorolni rajta.
- ☐ A tárgyat enélkül is teljesíthetőnek tartottam.
- ☐ Nem volt időm gyakorolni a tárgyra.
- ☐ Más tárgyakba befektetett időt jobban megtérülőnek tartottam.
- ☐ A felület nehezen volt használható.
- ☐ Nem láttam értelmét a használatának.
- ☐ Egyéb:

Ha bármilyen egyéb megjegyzése, javaslata lenne, kérjük írja az alábbi területre (nem kötelező):

További szöveges visszajelzésre lehetőséget biztosítunk online felületen keresztül (akár anonim módon, akár névvel).

Köszönjük szépen a segítségét!

B. függelék

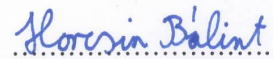
Mesterséges intelligencia használata

NYILATKOZAT

Kijelentem, hogy mesterséges intelligencia alkalmazást/programot (pl. képalkotó, szövegalkotó) nem használtam a dolgozat készítése során.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem, illetve mesterséges intelligencia alkalmazás/program segítségével vettem igénybe vagy a dolgozattal kapcsolatban szerzői jogi jogsértés ténye merül fel, a Szekció megtagadja a dolgozat befogadását.

Dátum: 2024. június 30.



Aláírás

Horcsin Bálint
Név (olvashatóan)

Irodalomjegyzék

- Anjali, Caraza-Harter, T., & Swift, M. M. (2020). Blending containers and virtual machines: a study of firecracker and gVisor. *Proceedings of the 16th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, 101–113. o. <https://doi.org/10.1145/3381052.3381315>
- Chandra, B., Joseph, M., Radhakrishnan, B., Acharya, S., & Sudarshan, S. (2016). Partial marking for automated grading of SQL queries. *Proc. VLDB Endow.*, 9(13), 1541–1544. o. <https://doi.org/10.14778/3007263.3007304>
- Haldeman, G. (2021). Automated feedback generation for programming assignments. <https://rucore.libraries.rutgers.edu/rutgers-lib/66902/PDF/1/play/> (2024.04.16.)
- Horcsin, B., & Abonyi-Tóth, A. (2023). Novel database manager agnostic method for automatically checking the correctness of a submission for a database management task. S. Elżbieta & Z.-C. Katarzyna (Szerk.), *Aktualne wyzwania współczesnej edukacji* (126–141 o.). Uniwersytet Radomski im. Kazimierza Pułaskiego.
- Horváth, G., Horváth, G., & Zsakó, L. (2017). A BÍRÓ és a MESTER – az online értékelés szerepe a programozás oktatásban. K. Krisztina & H. Zoltán (Szerk.), *Mérési és értékelési módszerek az oktatásban és a pedagógusképzésben* (89–102 o.). ELTE Eötvös Kiadó. https://www.eltereader.hu/media/2017/07/Diszciplinak_5_READER.pdf (2024.04.13.)
- Kádek, T., Kósa, M., Orosz, A., & Tóth, R. (2017). Új fejlesztési irányok programozási feladatok megoldáskiértékelő rendszereihez. <http://konferenciak.inf.elte.hu/infodidact/InfoDidact17/Manuscripts/KTKMOATR.pdf> (2024.04.16.)
- Király, S., Balla, T., & Király, R. (2022). Learning SQL by practicing on popular movie databases. *Central-European Journal of New Technologies in Research, Education and Practice*, 4(1), 16–24. o. <https://doi.org/10.36427/CEJNTREP.4.1.4465>

- Korom, R. (é. n.). *Adatbázis modellezés és SQL feladatok automatikus kiértékelése gráf-elméleti alapokon*.
https://www.inf.u-szeged.hu/sites/default/files/koromrichard_tdka2020o.pdf
 (2024.04.13.)
- Koyya, P., Lee, Y., & Yang, J. (2013). Feedback for Programming Assignments Using Software-Metrics and Reference Code. *ISRN Software Engineering, 2013*, 1–8. o.
<https://doi.org/10.1155/2013/805963>
- Kumar, D., & Mishra, K. (2016). The Impacts of Test Automation on Software's Cost, Quality and Time to Market [Proceedings of International Conference on Communication, Computing and Virtualization (ICCCV) 2016]. *Procedia Computer Science, 79*, 8–15. o. <https://doi.org/10.1016/j.procs.2016.03.003>
- Mareš, M. (2021). Security of Grading Systems. *OLYMPIADS IN INFORMATICS*, 37–52. o. <https://doi.org/10.15388/ioi.2021.04>
- Oktatási Hivatal. (é. n.). Kétszintű érettségi – Publikus statisztikák.
<https://www.ketszintu.hu/publicstat.php> (2024.04.14.)
- Pánovics, J., Kádek, T., & Biró, P. (2022). A ProgCont rendszer jelene és jövője.
<https://people.inf.elte.hu/szlavi/InfoDidact22/Manuscripts/PJKTBP.pdf>
 (2024.04.16.)
- Sadiq, S., Orlowska, M., Sadiq, W., & Lin, J. (2004). SQLator: an online SQL learning workbench. *ACM SIGCSE Bulletin, 36*, 223–227. o. <https://doi.org/10.1145/1026487.1008055>
- Shah, B., & Pareek, J. (2022, április). Automated Evaluation of SQL Queries: Eval_SQL.
https://doi.org/10.1007/978-981-16-6616-2_9
- Siegler, G. (2014). Érettségi vizsgatárgyak elemzése – 2009-2012 tavaszi vizsgaidőszakok – Informatika.
https://www.oktatas.hu/pub_bin/dload/unios_projektek/tamop318/erettségi_vizsgatargyak_elemzese/informatika.pdf (2024.04.14.)
- Skalka, J., Drlík, M., & Obonya, J. (2019). Automated Assessment in Learning and Teaching Programming Languages using Virtual Learning Environment, 689–697. o. <https://doi.org/10.1109/EDUCON.2019.8725127>
- Task Management System. (é. n.).
<https://tms-elte.gitlab.io/> (2024.05.03.)
- Wang, Y., Mäntylä, M. V., Liu, Z., & Markkula, J. (2022). Test automation maturity improves product quality—Quantitative study of open source projects using continu-

ous integration. *Journal of Systems and Software*, 188, 111259. o. <https://doi.org/10.1016/j.jss.2022.111259>