SZAKDOLGOZAT

Szabóné Balla Henrietta

Debrecen, 2007

Debreceni Egyetem

Informatika Kar

Számítógéppel támogatott értékelés az oktatásban

Témavezető:
Dr. Nyakóné dr. Juhász Katalin
tudományos főmunkatárs

Készítette:
Szabóné Balla Henrietta
informatika tanár szakos hallgató

Debrecen, 2007

Tartalomjegyzék

1. Bevezetés

A szakdolgozat fő témája az oktatásban alkalmazott, számítógéppel segített számonkérés. A dolgozat 3 fő részből tevődik össze, az utolsó témakört egy általános és egy konkrét részre tagoltam.

Az első részben az értékelés didaktikai hátterét részletezem. Taglalom az értékelés folyamatát, kezdve a típusaival, csoportjaival. Ismertetem a tudásmérés során alkalmazott feladatok típusait, az egyes típusok előnyeit, hátrányait, a feladatlapok előállításának fontos szempontjait. Az első szerkezeti rész tartalmazza még az eredmények kiértékelésének folyamatát, érdemjegyre váltását is.

A második rész a számítógépek oktatásban elfoglalt helyéről szól. Vázolom a számítást segítő szerkezetek fejlődési szakaszait az ókortól egészen napjainkig. Részletesebben tárgyalom ezen belül a számítógépekkel segített vagy éppen vezérelt oktatást, és a segítségükkel előállított feladatlapok készítését, kiértékelését.

Dolgozatomban a legnagyobb hangsúlyt a gyakorlati megvalósítás bemutatására fektettem. A harmadik részben bemutatom általánosan a gyakorlati részhez felhasznált PHP nyelvet, a HTML-t, a MySQL-t. Ezután részletesen leírom, hogy hogyan készítettem PHP-ben egy számonkérő tesztet. Kezdem az adatbázis létrehozásával, aztán a HTML és végül a PHP kód részletezésével. A forráskód végeláthatalan taglalásába nem megyek bele, hanem kiemelem a program működése szempontjából fontos részeket.

Olyan tesztet készítettem, amelybe kétféle módon lehet belépni, oktatóként (admin), és tanulóként. Az oktató létrehozhat új felhasználót (tanulót), lekérheti a diákok teszteredményét, és törölheti azokat. A tanuló belépés után egyszer elvégezheti a tudásmérést és annak végeztével azonnal megkapja az eredményt. Az eredmény egy adatbázisba kerül. A számonkérés addig nem ismételhető meg, amíg az oktató ki nem törli az eredményt az adatbázisból.

2. Az értékelés didaktikai alapjai

A tanulók munkájának értékelése a tanulási-tanítási folyamat fontos részét képezi. A pedagógiai értékelés [3] információt ad a követelmények minőségéről, visszajelzést ad a tanulás folyamata számára, és méri a kimeneti teljesítményt. A keletkező információ elsősorban a diák tanulási folyamatában hasznosul. Alkalmazza a diák, aki közvetlenül visszacsatolja az információt, és a tanár is, aki a diák tanulásának irányításában használja fel a keletkezett információt.

Az értékelés viszont nem csupán egy információ, hanem magába foglalja a más eredményekhez, teljesítményekhez történő viszonyítást is. Az értékelés az informáló funkcióját akkor tölti be eredményesen, ha objektív, vagyis ha egységes, fix követelményhez viszonyítunk, és tárgyszerűen minősítünk.

Az értékelésnek motiváló hatása is van. Régóta bizonyított tény a pszichológiában, hogy a tanári értékelés kapcsolatban áll a tanulói teljesítménnyel. Az értékelés főként maga is pozitív vagy negatív megerősítés a tanuló számára. Viszont az értékelésnek a diák környezetéből kiváltott hatása sem elhanyagolható.

Az értékelés az informáló funkcióját akkor tölti be eredményesen, ha szubjektív, vagyis ha egyéni, személyre szóló, lehetőleg tartalmazzon megoldási lehetőséget fejlesztésre és korrekcióra.

2. 1 Az értékelés típusai

Leíró értékelés:

Tényszerű jelentés a nyújtott teljesítményről. Nem tartalmazza a minősítés mozzanatát, ezért szó szoros értelemben nem is értékelés. Mégis azért tartozik ide, mert előre meghatározott szempontok alapján értékel.

Normatív, normára vonatkoztatott értékelés:

A teljesítményt egy meghatározott normához viszonyítja, amely mások teljesítményén alapul. Az iskolai teljesítménymérés általában norma alapú, a gyenge és a jó tanulók megkülönböztetésére szolgál.

Kritériumorientált értékelés:

A célokhoz való viszonyítás jelenik meg, szemben a normatív értékeléssel. Nem a nyújtott, hanem az elvárható teljesítményből, követelményekből, a tanulás céljából indul ki. Ehhez szükséges a tudás vizsgálata, kritériumokra bontása. A kritérium a tudás olyan elemi egysége: amelyhez feladat illeszthető, és a feladat megoldása egyértelműen meghatározza, hogy teljesített a kritérium, vagy sem. Az értékelés eredménye a megfelelt vagy a nem megfelelt minősítés.

Standardra vonatkoztatott értékelés:

A kritériumorientált értékelés az alapja. Az elvárható tudásból indul ki. A kritériumokat hierarchiába rendezik. A teljesítményfokozatokhoz minőségi skálát rendelnek. A skála, a kritériumok és maga az értékelési eljárás alkotják a standardot.

2. 2 Az értékelés csoportjai

Az értékelésnek három csoportját különböztetjük meg az oktatásban betöltött szerepe szerint. Az értékelések csak akkor tudják jól ellátni a feladatukat, ha megfelelő időben alkalmazzák őket. Az egyes értékelési típusok alkalmazása között különös gondot kell fordítani a megfelelő arányok betartására.

Diagnosztikus értékelés:

Az oktatási folyamat egy szakasza előtt szükséges. Célja felmérni az aktuális állapotokat, a tanulók elsajátított tudását. A tanárok ennek eredményeihez képest alakíthatják ki a tanítási stratégiát, figyelembe véve a szóban forgó osztály erős és gyenge pontjait. A diagnosztikus értékelés a tantárgyi tanítás-tanulás, és a képességfejlesztés szabályozásában, a tennivalók tervezésében egyaránt hatékonyan segítheti a pedagógusok munkáját. Diagnosztikus értékelés során nem célszerű számszerűsíteni a tudást, az osztályzat miatti aggodalom befolyásolhatja az eredményt. A tényleges, konkrét tudás felmérése a cél, éppen ezért célszerű az értékelést több alkalommal, más helyzetekben elvégezni, mert külső okok esetenként eltorzíthatják az eredményt.

Formatív értékelés:

Az oktatási folyamat közben ajánlatos. A formatív értékelés alapvető szerepe a tanulási-tanítási folyamat irányítása, a tanuló teljesítményéről adott visszacsatolás. Javítja a hiányos tananyagtartalmak elsajátítási módját. Többféle formája lehet: szóbeli, írásbeli számonkérés, tesztek, rendszeres részterületekre irányuló kontrollvizsgálat. A formatív értékelés akkor hatékony, ha rendszeresen alkalmazzák, ha objektív, minden egyes tudáselemre kiterjed, felméri a tanuló egyéni fejlődését, rávilágít a tanuló erősségeire és hiányosságaira. Így részletesen tájékoztatja a tanulókat és a tanárokat a hiányosságokról. Kutatások szerint a formatív értékelés jó hatással van a tanulói teljesítményre, aktívabbá teszi a diákokat a tanórán, változatos tanítási módszereket tesz lehetővé. A formatív értékelési mód nem lehet a diák teljesítményét osztályzattal minősítő, azaz nem függhet tőle a tanuló iskolai előmenetele, mivel ennek célja az eredményesebb tanulás támogatása, segítése.

Szummatív (összegző) értékelés:

A tanulási folyamat egyes szakaszainak végén segít ellenőrizni az elért teljesítményszinteket, a tantervi követelményekből kiinduló, pontos kritériumrendszeren alapuló értékelés. A szummatív értékelés alapkövetelményei, hogy korrekt, nyilvános, érvényes, megbízható, objektív legyen. Alkalmasnak kell lennie minősítésre, szelekcióra. Egységes, pontosan meghatározott követelmények alapján minden tanulóra legyen érvényes, azonos mércével mérje a tanulók tudását. Célja a tanulók közötti különbségek feltárása, a teljesítmény érdemjeggyé történő alakítása, annak rávilágítása, hogy a tanulók milyen szinten tudták elsajátítani az anyagrészt, megmutatja, hogy hol szükséges differenciált fejlesztő beavatkozás. Ehhez az értékeléshez tartozik az osztályozás és a vizsgáztatás kérdése. A tanulót teljesítménye alapján sorolják kategóriákba, az alapján, hogy az elvárásoknak milyen mértékben felel meg a tanuló teljesítménye. A szummatív értékelés a pedagógiai munka fontos irányító eszköze lehet, irányadó lehet a tanárok, és a tanulók számára is. Elsősorban a tanárok munkáját segíti, mivel visszajelzést ad a pedagógiai munka folytatására nézve.

2. 3 Feladatlap készítése

A továbbiakban az értékelés különféle eszközei közül a feladatlap csoportjait ismertetem.

Formai szempontból a feladatok lehetnek: feleletválasztásos (zárt) feladatok, kiegészítéses feladatok, feleletalkotó (nyílt) feladatok.

Feleletválasztásos (zárt) feladatok:

A feladatlapokon, teszteken alkalmazott olyan feladat típus, amelynél a tanuló előre megfogalmazott válaszalternatívákból választja ki a helyénvalót. A feleletválasztásos feladatok közös tulajdonsága a célirányosság, a reprodukálhatóság, az objektivitás. Előnye, hogy a legkönnyebben és leggyorsabban lehet javítani, és lehetőség van hibaelemzésre is. Hátránya, hogy lehetséges, hogy éppen a helytelen válasz rögzül a tanuló emlékezetében, nagy teret enged a találgatásoknak, és elkészítésük munkaigényes.

1. Választásos feladatok: Az alábbiak szerint a helyes vagy helytelen válaszokat kell megfelelő számban megjelölni.

   1.1. Egyszerű választás: A lehetséges válaszalternatívákból a jó megoldást kell megjelölni. A helyesként megjelölt válaszok számától függően lehet egyszeres vagy többszörös választás.

   1.2. Hibakutatás („Error Correction”): A lehetséges válaszalternatívákból a rossz választ kell kiválasztani. Egy és több rossz válasz megjelölése kérhető. A feladattípust főként a nyelvoktatásban használják.

   1.3. Többszörös választás: A válaszok kombinációi képezik a helyes megoldást. Könnyű és objektív javítási lehetőség jellemzi. Képes a tudás szintjeit is mérni, viszont nehéz jó feladatlapot készíteni.

   1.4. Relációanalízis („Relation Analysis”): Kijelentések helyességét kell eldönteni. A kijelentések összetett mondatok, melyeknek első része egy állítás, második része pedig egy indoklás. A két mondatrész helyességéről vagy helytelenségéről, és egymáshoz viszonyított kapcsolatáról kell dönteni a

válaszadás során. A felsorolt szabványos lehetőségek közül kell kiválasztani az egyetlen helyes megoldást.

2. Alternatív feladatok („True-False Question”): A feladatok során két alternatíva közül kell választani. A válasz vagy helyes, vagy helytelen. Hatékony nagy mennyiségű tananyag számonkérésére. Gyors és objektív pontozást tesz lehetővé. Hátránya, hogy nagy teret enged a találgatásnak.
3. Páros asszociáció, válaszok illesztése, csoportosítása: Egymáshoz tartozó fogalmakat kell összekötni. Viszonylag könnyű megszerkeszteni, objektív javítást tesz lehetővé. Leginkább egyszerű képességek mérésére alkalmas.

   3.1. Egy az egyhez típus

   3.2. Egy a többhöz típus

Kiegészítéses feladatok:

Formailag a feleletalkotó feladatokhoz állnak közelebb, viszont bizonyos megkötésekkel a feleletválasztós feladatokhoz hasonlítanak leginkább. A kiegészítéses feladatokban a feleletalkotás módja a nem teljes válasz. A keresendő válasznak mindig az ismeret lényeges részének kell lennie. Formája szerint a kiegészítés szöveges és rajzos is lehet, és ezek kombinációi: szöveget szöveggel, szöveget rajzzal, rajzot szöveggel, rajzot rajzzal kiegészítő válasz.

1. Egyszerű kiegészítés: Egy szövegből szavak vannak kihagyva. A cél olyan szavak találása, amelyekkel helyreállítódik az eredeti szöveg.
2. Kiegészítés rajzzal vagy jelöléssel
3. Táblázat kitöltése

Feleletalkotó, nyílt feladatok:

A befogadás folyamatának mérésére alkalmas írásbeli feladattípus. A tudás többféle összetevőjét lehet tesztelni, de javításuk, általában több figyelmet igényel, nehezebben automatizálható, és az objektív javítás nagyon nehéz.

1. Rövid választ igénylő feladatok: a részletes követelményekben meghatározott tartalmak felismerését, megnevezését, meghatározását jelentik. Szavak, számok

megadása. Lecsökkenti a találgatások kockázatát, viszonylag könnyű elkészíteni, a javítása objektív.

2. Hosszú választ igénylő feladatok: Hosszabb kifejezések, egész mondat megadása. Lehet elemző vagy összehasonlító dolgozat. Javítása szubjektív, és nehézkes.
3. Esszé: az előző típusú feladattól abban tér el, hogy a tartalmon felül a szerkesztettséget, a stílust, a válasz kerekségét is értékelik. Több mondatból álló összefüggő szöveg írása. Gyorsan és könnyen lehet összeállítani, kizárja a találgatást, viszont javítása sok időbe telik és teljes mértékben szubjektív.

A feladatírás precíz nyelvhasználatot igényel, az utasításnak egyértelműnek kell lenni. Egy témakörhöz tartozó feladatot egy csoportba tanácsos szerkeszteni. A feladatok célja azt kideríteni, hogy a tanulók mit tudnak, és nem azt, hogy mit nem.

2. 4 Eredmények kiértékelése

A feladatlap készítés magába foglalja a hozzá tartozó megoldókulcs elkészítését, ponthatárok meghatározását. Egy feladatlapnak eltérő nehézségű feladatokat kell tartalmaznia. A könnyebb feladatokra kevesebb, a nehezebb feladatokra több pontszám adható.

Azt az eljárást, amelynek során a különböző részfeladatokhoz pontszámokat rendelünk, súlyozásnak nevezzük. A tanuló a helyesen megoldott feladatra az egész pontszámot megkapja. Ha több helyes válasz esetén nem mindegyiket jelöli meg, akkor a megoldásra általában nem adunk pontot. A feladatok súlyozásánál ezt figyelembe kell venni.

A súlyozásnak két változata van.

Empirikus súlyozás

Egy feladat nehézségét az alapján állapítjuk meg, hogy milyen eredményt ér el a csoport egésze. Az elért eredményeket egymáshoz szeretnénk hasonlítani. A javítás során azt kell vizsgálni, hogy a tanulók hány százaléka oldotta meg jól a feladatot, és csak ezután kell megállapítani a pontokat. A kapott százalékokat kivonjuk 100-ból, a különbséget osztjuk 10-zel, majd az eredményt egészre kerekítjük. Ezek az egész számok az úgynevezett empirikus súlyok. Így több pontot érnek azok a feladatok, amelyeket

kevesebben oldottak meg. A könnyebbnek bizonyuló feladatok kevesebb pontot kapnak.

Hiba lehet, ha a feladat van rosszul megfogalmazva, ez megmagyarázza a rossz megoldásokat, és akkor lehet, hogy éppen a „jó” megoldások a hibásak, amire a sok pontszám jár.

Itemes súlyozás

A dolgozatjavítás első lépéseként a feladatokat részfeladatokra bontjuk, a feladatok legkisebb önálló egysége az item. Minden egyes itemhez egy egész számot rendelünk. Ezek közül a legegyszerűbb itemhez 1 itemsúlyt rendelünk, és a többi itemet ehhez képest súlyozzuk.

A találgatások, és ezáltal a véletlenül szerzett pontok elkerülése végett szokták alkalmazni az úgynevezett amerikai vagy negatív pontozást. E szerint a módszer szerint a helytelen megoldásokat büntetni kell negatív pontszámokkal. A legáltalánosabb megoldás szerint a helyes megoldás 1 pont, a helytelen -1 pont, 0 pont jár a válasz hiányáért. Ezt mindkét irányban lehet csiszolni úgy, hogy a jó megoldásért több pozitív pontot, a rossz válaszért pedig több negatív pontot adunk. Hátránya a módszernek, ha elvétjük az egyensúlyt és a rossz válaszokért túlságosan sok pontot vonunk le. Így fennáll az esélye annak, hogy még ha tudja is a tanuló a helyes választ, fél bejelölni. Sok elemző véleménye szerint szükségtelen az amerikai pontozás egy megfelelően elkészített feladatlapnál, mivel az kivédi a találgatásokat. A jó feladatlap teljesen lefedi a tananyagot, különböző nehézségű feladatokat tartalmaz, a kérdések megfogalmazása érthető és egyértelmű, mindenre felhívja a figyelmet. Egy feladatlap csak akkor méri minden tanuló teljesítményét, ha valamilyen szinten a csoport minden tagja számára megoldható.

Az értékelés utolsó mozzanata az érdemjegy kiszámítása. Az érdemjegy (n) megállapítása a következő képlettel számítható:

$$n = \frac{\sum_i p_i \cdot r_i}{\sum_i p_i},$$

ahol p_i az *i*-edik feladat pontszáma, r_i értéke 1 vagy 0 aszerint, hogy helyes vagy helytelen az *i*-edik feladat megoldása.

3. Számítógépek az iskolában

Az oktatás során számos tanítást segítő eszközt használhatunk fel. Jelenleg az eszközök száma szinte megszámlálhatatlan. A technika fejlődésével újabb és újabb kapuk nyílnak meg az oktatás terén is. A legmodernebb elektronikus vívmányok is teret kapnak a tanítás során. Napjainkban használt eszközök: számítógép, digitális tábla, interaktív tábla, flipchart tábla, diavetítő, írásvetítő, magnetofon, televízió, CD lejátszó, oktató CD-k megszámlálhatatlan sokasága, videomagnó, DVD lejátszó, projektor, stb. Az internet is rengeteg információt nyújt a tanuláshoz.

3. 1 Oktatást segítő eszközök a kezdetektől

Kiemelném a számítógép és az internet szerepét, mely egyre nagyobb szerepet kap az oktatásban.

A számítógép

A pontos, gyors számítások elvégzése a kezdetektől központi szerepet játszott a tanításban. Így minden korban arra törekedtek, hogy egyre pontosabb és gyorsabb számításokra alkalmas eszközöket fejlesszenek ki. Ezek az eszközök aztán a hétköznapi életben is alkalmazhatónak bizonyultak, számos felhasználási lehetőségük terjedt el. Mivel az élet különböző területein is meghonosultak ezek az eszközök, ezért a későbbiek során nem csak a tanulást segítették, hanem maguknak az eszközöknek a használatát is tanítani kellett.

A tanítást segítő eszközök az ókortól napjainkig nagy utat tettek meg. Az első abakuszt ie. 3000 körül használták számítások segítésére. Ez az eszköz a világ minden táján elterjedt és a mai napig megtalálhatók korszerűsödött változatai, a különböző golyós számolóeszközök. Pontosabb műveleteket végezni velük azonban vagy lehetetlen volt vagy csak nagyon lassú, ezért fontos tudományos felfedezések történtek a mechanikus számológépek feltalálásától a számítógépek kifejlesztéséig.

1588 Jost Bürgi elkészíti az első logaritmustáblákat. 1622-ben William Oughtred alkalmazott először logaritmus skálát a két egymáson elcsúsztatható vonalzókon. 1650-ben Pattridge elkészítette az első mai formájú logarlécet.

1623 Wilhelm Schickard egyszerű négy alapműveletes masinát készít. 1642 Blaise Pascal számológépet készít, amely sorozatgyártásban is készül.

1672 Gottfried Wilhelm Leibnitz mechanikus számológépet készít, tulajdonképpen Pascal gépét fejleszti tovább. Ezzel a géppel már szorozni, osztani és gyököt is lehettet vonni. Leibnitz nevéhez még két fontos felfedezés is fűződik. 1666-ban bebizonyítja, hogy egy számolási művelet egymás után elvégezhető egyszerű, elemi lépések sorozatára bontható. 1679-ben ismerteti a kettes számrendszert.

1810 Joseph Marie Jacquard lyukkártya vezérlésű szövőgépet állít elő. A lyukkártyát a későbbiekben számítógépeknél is alkalmazzák.

1820-as évek elején Charles Babbage megtervezi a Difference Engine, amely logaritmus táblázatok pontos és gyors elkészítését teszi lehetővé. 1833 Babbage megtervezi az Analitical Engine-t, ami anyagi okok miatt nem készül el. 1885 Stevens Borroughs elkészíti az első billentyűzettel, nyomtatóval ellátott összeadó-gépet. 1886 - ban Herman Hollerirth feltalálja a Lyukkártya-feldolgozó gépet elektronikus számlálásra.

1939 : ABC - az első digitális számítógép, amelyet Dr. John Astanasoff tervezett.

A budapesti születésű Neumann János 1946-ban megfogalmazza elveit a számítógépről:

- elektronikus elven működő számítógép
- tárolt program elve, legyen belső memória is
- kettes számrendszer használata
- soros utasítás végrehajtásának elve
- központi vezérlőegység alkalmazása
- aritmetikai egység alkalmazása

A számítástechnika korszaka hivatalosan 1951. június 5-én kezdődött, amikor az első UNIVAC-ot leszállították az Egyesült Államok Népszámlálási Hivatala számára. Ez volt az első kereskedelmi forgalomban elérhető számítógép.
1943-ban fejlesztették ki az első elektronikus számítógépet, és az 1958-ban előállított számítógépeknél már második generációs számítógépekről beszélhetünk a tranzisztorok megjelenése miatt. 1965-től kezdve harmadik generációs gépek jönnek létre, ekkor alkották meg az integrált áramköröket. A negyedik generációs számítógépek 1971-től

jelentek meg, amikor az Intel piacra dobta a 4004-es processzort. A mikroprocesszor köré épített személyi számítógép rohamosan elterjed, ez a folyamat ma is tart. Általánossá vált használatuk szövegszerkesztésre, táblázatkezelésre, grafikára, adatbáziskezelésre, stb.

Az internet

A világháló [9] gyakorlatilag sok-sok egymástól független hálózatok összessége, egymáshoz való kapcsolódása. Az Interneten nincs központi gép. A hálózatra kötött összes gép egyszerre fő- és al-állomás. Az Internet, számítógépek és telefonvonalak olyan elméleti szerveződése, amelynek bármely pontja képes kapcsolatot teremteni bármely másik pontjával. A 80-as évek elején az első PC-k megjelenésével létrejöttek a helyi hálózatok, és gombamód szaporodni kezdtek. Az Internete elődjét az ARPANet-et az Advanced Research Projects Agency fejlesztette ki 1969-ben. Az eredeti cél egy olyan hálózat létrehozása volt, amely lehetővé teszi, hogy a kutatói számítógépek az egyetemek között kommunikálhassanak egymással. Az Internet adatforgalma manapság óriási. Az internetnek számos alkalmazása létezik, pl.: keresőgépek, elektronikus levelezés, webmail, hírlevél, dinamikus HTML, internet-telefon, stb.

A keresőgépek a legnépszerűbb és leghasznosabb Web-szolgáltatások, amelyek segítik az Interneten való könnyebb eligazodást, rengeteg témakörben kereshetünk információkat például a munkához, tanuláshoz. Segítségükkel könnyedén megtalálhatunk különböző dokumentumokat, képeket, programokat.

3. 2 Számítógép az oktatásban

A számítógép megjelenhet, mint az oktatás szervezője [7]. Előre felépített sorrend szerint irányítja az oktatás menetét. Tananyagot nem tárol, csak a tanulók eredményeit, a tanulók nem állnak közvetlen kapcsolatban a számítógéppel. Viszont, ha mint oktatási eszköz jelenik meg a számítógép, akkor nem csak irányítja a tanulás folyamatát, hanem nagy mennyiségű tananyagot tárol, gyakorló feladatokat generál, feladatlapot értékel ki, ellenőrzi a válaszokat, eredményt számol, eltárolja az eredményeket.

Persze használhatjuk a számítógépet mindössze számonkérésre vagy gyakoroltatásra is. Segítségül szolgálhat mind a tanároknak, mind a diákoknak. A számítógéppel segített értékelés típusai azonosak a hagyományos eszközökkel végzett értékelés típusaival.
A számítógéppel végzett értékelés előnyei:

- mint láttuk általánosan is, a feladatlapok (tesztek) objektívan értékelhetők,
- javításuk pontos és gyors,
- gyors visszacsatolást biztosítanak,
- többfajta tanulói teljesítmény mérhető,
- az eredmények tárolhatók,
- feladat adatbázisból véletlen kiválasztás alapján adható kérdés.

Hátrányok:

- a jó feladatlapok készítése időigényes munka,
- nagy a meghibásodás kockázata, mind hardver, mind szoftver szinten,
- számítógép használati előismeretekre van szükség, mind az oktatónak, mind a tanulóknak,
- használatuk költséges, mivel megfelelő számú számítógépre van szükség.

A számítógép kiválóan alkalmas olyan órákon történő felhasználásra is, amelyeken valamilyen modellt kell bemutatni, vagy eseményt kell szimulálni. Ilyen tantárgy például a biológia, kémia, fizika. Ebben az esetben a számítógéppel történő megjelenítés részletesebb lehet, jobban nyomon követhető, érthetőbb, a keletkező adatok gyorsabban, pontosabban és hatékonyabban kerülhetnek feldolgozásra. A bemenő adatok módosításával figyelemmel lehet kísérni az eredmények változását. Hátrányként itt is elmondhatjuk, hogy csak akkor lehet megfelelő hatékonysággal használni a számítógépet erre a célra, ha a tanulók vagy a tanár kellő ismerettel rendelkezik a számítógép működtetésére, kezelésére vonatkozóan.
A számítógép tehát egyre inkább átveszi más, az oktatásban alkalmazott eszköz szerepét, de nem mindet helyettesítheti. Nem tudja átvenni az élőben elvégzett kísérletek hangulatát, élményét.
Manapság egyre több oktató CD kerül forgalomba, ami nem csak az iskolai oktatást, hanem az otthoni felkészülést is segíti. Otthon segítséget nyújthat a számítógép

gyakorlásban, különböző adatbázisok, táblázatok elkészítésében, dolgozat írásában, tananyaghoz végzett kutatómunka szerkesztésében, különféle görbék előállításában, stb.

4. Általánosan a HTML-ről, a PHP-ról és a MySQL-ről

4. 1 HTML

A HTML (Hyper Text Markup Language) web oldalak készítéséhez kifejlesztett nyelv.

A 60-as években Ted Nelson előállt a hipertext formátum ötletével. A 80-as évek elején létrejönnek a számítógépes hálózatok, és ezzel egyidőben megfogalmazódik az igény arra, hogy hardver- és szoftverfüggetlen formázott szövegeket lehessen létrehozni. Az IBM ki is fejleszt egy ilyen pszeudo programozási nyelvet, a GML-t. Ebből a Nemzetközi Szabványügyi Hivatal megteremti az SGML-t. A HTML az SGML-nek egy leszűkített változata, amelyet 1990-óta használnak.

A HTML dokumentum normál szöveges állomány, amely bármely egyszerű szövegszerkesztővel létrehozható, illetve módosítható. Ebben a fájlban találhatóak a nyers szövegek és a szimbólumok, jelek (tag-ek), amik megmondják a programnak, hogy jelenítse, formázza meg, hogyan dolgozza fel a fájl tartalmát.
Amikor egy HTML fájlt egy böngészőprogrammal (pl.: Microsoft Internet Explorer, Firefox, Mozilla) megnézünk, akkor először a web lap (fájl) a távoli számítógépről (szerverről) a mi számítógépünk memóriájába kerül. Ezután a gépünkön lévő böngészőprogram a tag-eket megpróbálja megjeleníteni, olyan módon, ahogy az a HTML szöveges állományában le van írva. Gond merülhet fel olyankor, ha az állomány tag-jei újabb verziójúak a mi böngészőnknél.

4. 2 PHP

Széles körben használt általános célú szkriptnyelv, amely kifejezetten alkalmas - akár HTML-be ágyazott - webalkalmazások fejlesztésére [5]. A PHP és a MySQL valamelyik változata elengedhetetlenül szükséges, ha saját gépünkön szeretnénk web alkalmazásokat fejleszteni.

1995-ben elkészült a PHP elődje, a PHP/FI (Personal Home Page / Forms Interpreter), amely még csak néhányat tartalmazott a mai PHP alapfunkciói közül. Rasmus Lerdorf dolgozta ki. A PHP/FI akkoriban egy egyszerű Perl szkriptgyűjtemény volt, amely arra

szolgált, hogy nyomon kövesse az online önéletrajzához való hozzáféréseket. Kezdetben Perl-szerű változókat, űrlapváltozók automatikus értelmezését, HTML-be ágyazott szintaxist tartalmazott. Ezután a PHP/FI-nek több változata látott napvilágot.

Az első olyan verzió, amelyik jobban hasonlít a ma ismert PHP-re, a PHP 3.0 volt, amelyet Andi Gutmans és Zeev Suraski készített, és hivatalosan 1998 júniusában adták ki. Ez egy teljes újraírása volt a PHP-nek. Az egyik legnagyobb erősségét a PHP 3.0-nak a kiterjeszthetősége jelentette. Kedvező tulajdonsága, hogy a felhasználókat sok különböző adatbázist támogató infrastruktúrával, protokollokkal és API-kkal szolgálja ki. Jellemző még rá az objektum-orientált szintaxis támogatása és a sokkal erősebb és konzisztens nyelvi szintaxis.

2000-ben megjelent a PHP 4.0, mely a bonyolult alkalmazások hatékonyságának javítását tűzte ki célul. A 4-es verzió tartalmazta még több webszerver támogatását, HTTP session-öket, kimenet-pufferelést, a felhasználói inputok sokkal biztonságosabb kezelését és számos egyéb nyelvi konstrukciót.

A PHP 5 hosszú fejlesztés után 2004 júliusában jelent meg. Tartalmazza az új objektumorientált modellt és számos egyéb új szolgáltatást. Komplett, telepítővel rendelkező objektumorientált eszközkészlet.

Az különbözteti meg a PHP-t a kliens oldali nyelvektől - pl. JavaScript -, hogy a kód a kiszolgálón fut. A PHP gyors nyelv, amely abból adódik, hogy modulként illeszthetjük a webszerverbe, és használatakor a beállítások betöltése csak egyszer történik meg. A korábbi szerveroldali nyelveknél minden egyes használatnál betöltődtek a beállítások, amely a webszerver lassulásához vezetett.

A PHP használatával kis mennyiségű kódolással is egyszerű és hatékony kódokat írhatunk, illetve illeszthetünk weboldalainkba. Egy kód eredményét böngészőben megnézve, nem lehet megállapítani, hogy milyen kód állíthatta azt elő. Ezen felül a webszervert be lehet állítani úgy, hogy a PHP feldolgozzon minden HTML fájlt PHP blokkokat keresve, ezek után már tényleg nincs rá mód, hogy kitalálják, mit is rejt egy-egy program. A legjobb dolog a PHP használatában, hogy különösen egyszerű egy kezdő számára, de számos, fejlett szolgáltatást nyújt a professzionális programozó számára is.

A PHP-nek három fő felhasználási módja van:

- Szerver oldali programozás. Három komponens szükséges hozzá: a PHP értelmező, egy webszerver és egy webböngésző. A PHP program kimenetét a webböngészővel lehet megtekinteni, a szerveren keresztül elérve a szkriptet.
- Parancssori programozás. PHP programok szerver és böngésző nélkül is futtathatóak. Itt csak a PHP értelmezőre van szükség.
- Ablakozós alkalmazások írása. Viszont nem a legjobb nyelv grafikus felületű asztali alkalmazások írásához, de lehetőség van operációs rendszerfüggetlen programok írására.

A PHP használatakor szabadon választható az operációs rendszer és a webszerver. Ráadásul a függvény-alapú és objektum orientált programozás, vagy ezek keveréke közötti választás is lehetséges. Az egyik legjobb és legfontosabb tulajdonsága a nyelvnek az adatbázisok széles körű támogatása. Nagyon egyszerű adatbázisokat kezelő weblap készítése PHP segítségével.

A példaprogramomban a PHP által támogatott adatbázis a MySQL.

4. 3 MySQL

A MySQL [11] egy gyors, többfelhasználós, több szálú, SQL-alapú relációs adatbázis-kezelő szerver. Az MySQL az egyik legelterjedtebb adatbáziskezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú Rendkívüli gyorsasága és a webes alkalmazások hátthereként való alkalmazhatósága alapján joggal lett a legjobb rendszer-összeállítást jelképező rövidítés, LAMP alkotóeleme (e szó a Linux, Apache, MySQL és Perl vagy Python vagy PHP alkotta névegyüttes mozaikszava). Költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására. Az adatbázis-működtető szolgáltatások széles választékával látták el a fejlesztői, és gondoskodtak arról is, hogy ne terhelje túlságosan a központi egységet és a memóriát.

A fejlesztések 1996 körül kezdődtek el, amikor a szerzőknek TcX-nél egy nagy adatbázisokat biztonságosan és gyorsan kezelő SQL szerverre volt szükségük. Ekkoriban más adatbázis-rendszer fejlesztő cég nem rendelkezett ilyen környezethez igazán hatékony SQL szerverrel. Fontosságára utal, hogy a legfontosabb feladat

bármilyen alkalmazás létrehozásánál az alapos, átgondolt tervezés. A cél ismeretében legelőször az adatbázist kell megtervezni.

A ma használt MySQL tulajdonságai az alábbiak: beágyazott SELECT-ek, beágyazott adatbázis-könyvtár, az adatbázis-kezelőtől független tárolómotorok, mentési pontokat is kezelő tranzakciók, fejlett karakterkódolás támogatás, lehetőség megsérült tábláinak javítására, SSL támogatás, többszálas alkalmazások futtatása.

5. A tesztprogram

Első lépésben rátaláltam a www.apachefriends.org oldalon az XAMPP programra, amit aztán letöltöttem Windows XP gépemre a http://sourceforge.net/projects/xampp/ oldalról. Ezt a programot találtam legmegfelelőbbnek a tesztprogramom megírásához, mert tartalmazza többek között a következőket: Apache webserver, MySQL, PHP, Perl, FTP server, phpMyAdmin. A program telepítése után a gépem szerverként működik.
Szükségem volt még a MySQL GUI TOOLS-ra, amit a http://dev.mysql.com/downloads/gui-tools/5.0.html honlapról töltöttem le. A MySQL GUI TOOLS-on belül a MySQL Administrator és a MySQL Query browser könnyítette meg a programom készítését.

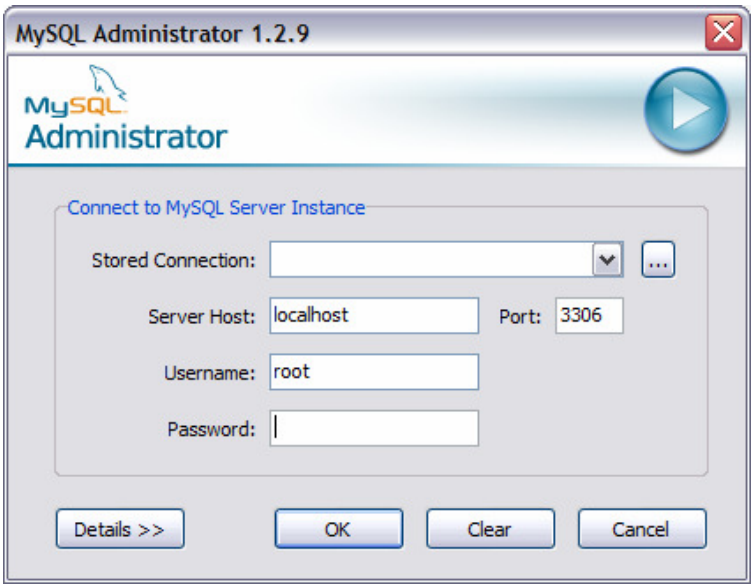
A **MySQL Administrator** egy MySQL-hez készült grafikus felhasználói felület, mely kényelmesebbé teszi a legkülönfélébb adatbáziskezeléssel kapcsolatos műveleteket. Maguk az adatbázisok, az azok eléréshez használható felhasználói azonosítók, valamint a jogosultságok kezelésében ad segítséget. Segítségével általános képet kaphatunk a szerver aktuális működéséről, lehetőséget biztosít akár újraindításra is, beállíthatjuk a szerver felhasználóinak jogait, stb. Különböző kijelzők adnak információt a kapcsolatokról, forgalomról, SQL lekérdezések számáról, memóriahasználatról, és egyéb mutatókról. Biztonsági mentésre és visszaállításra is lehetőséget ad az alkalmazás.

A MySQL Administrator főbb opciói:

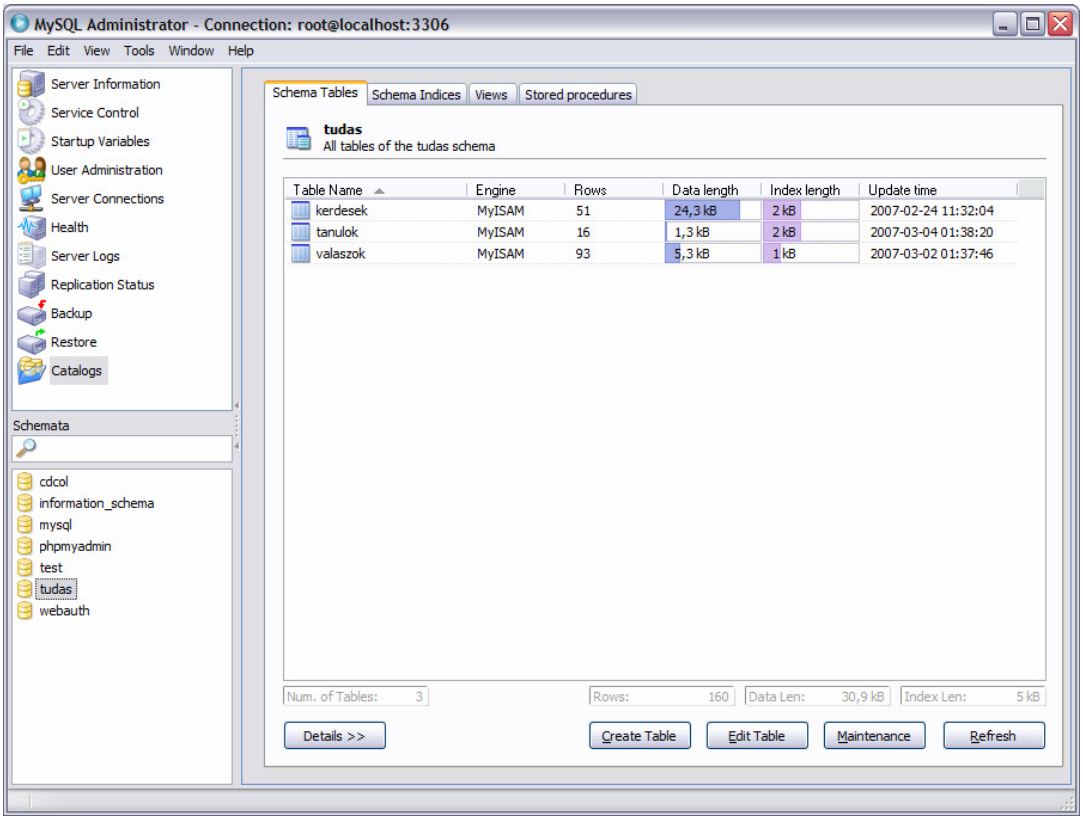
- Server Information: a kiszolgáló adatai.
- Service Control: a kiszolgáló szolgáltatás indítása vagy leállítása, illetve részletes beállítása.
- Startup Variables: a kiszolgáló indításakor alkalmazott beállítások, 9 főbb csoportba rendezve.
- User Administration: felhasználók áttekintése, egyes adatok szerkesztése.
- Server Connections: a kiszolgáló kapcsolatai az aktuális kapcsolatok, vagy a kapcsolódó felhasználók szerinti bontásban.
- Health: kapcsolatok, memória és változók figyelése.

- Server Logs: bejegyzések követése.
- Replication Status: a replikációs szerver adatai.
- Backup illetve Restore: adatbázisok mentése (akár automatikusan naponta vagy bizonyos napokon, hetente, havonta), számos beállítási lehetőséggel, illetve helyreállítása.
- Catalogs: adatbázisok adatai (táblák, azok mérete, sorok száma, indexek adatai).

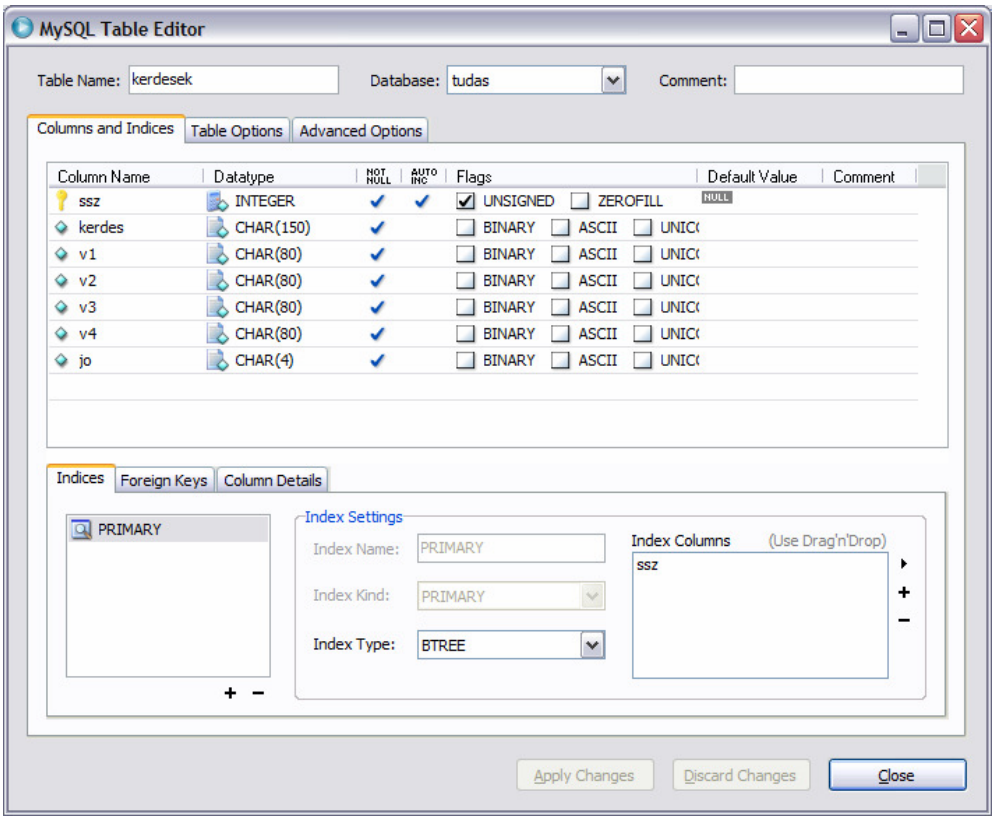
A szerver címének localhost-ot kell megadni, a felhasználónév root.

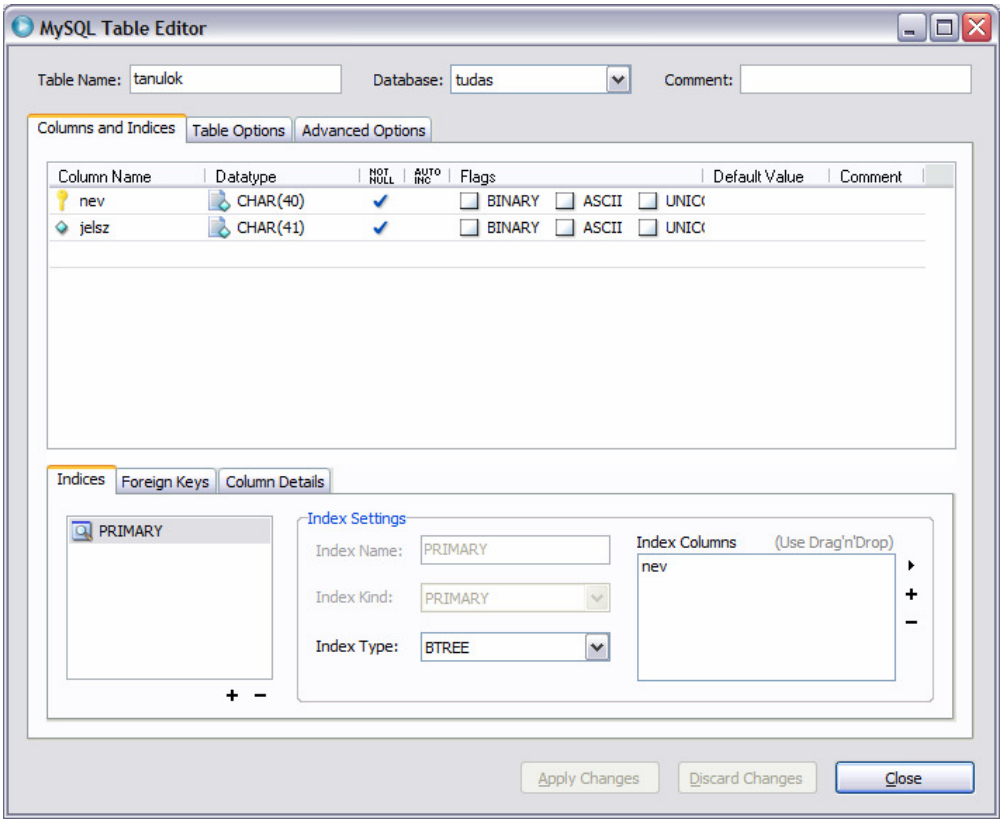


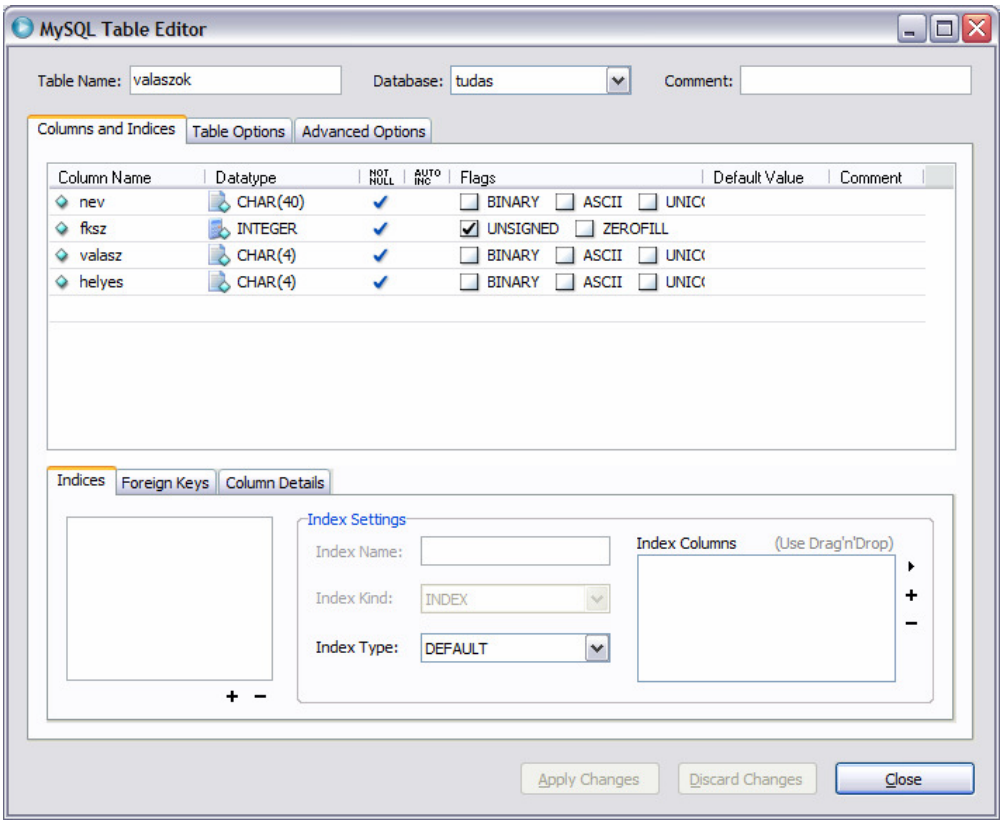
Létrehoztam egy új adatbázist „tudas” néven. A MySQL Administrator-ral is létre lehetne hozni az adatbázison belül a különböző táblákat, de azt célszerűbb a Query Browser-rel tenni.



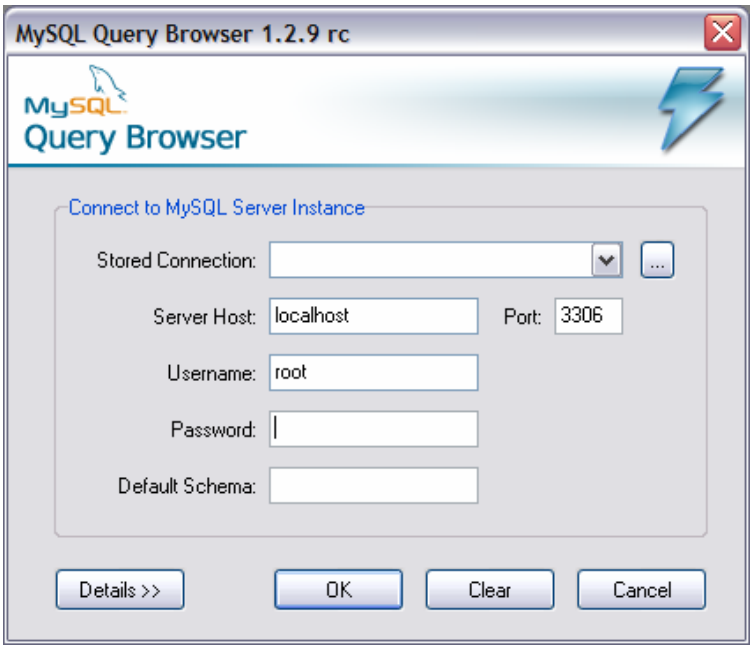
A három táblára vonatkozó adatokat is itt lehet beállítani.




MySQL Table Editor
Table Name: tanulok
Database: tudas
Comment:
Columns and Indices
Table Options
Advanced Options
Column Name
Datatype
NOT NULL
AUTO INC
Flags
Default Value
Comment
nev
CHAR(40)
BINARY
ASCII
jelsz
CHAR(41)
Indices
Foreign Keys
Column Details
PRIMARY
Index Settings
Index Name:
PRIMARY
Index Kind:
PRIMARY
Index Type:
BTREE
Index Columns
(Use Drag'n'Drop)
nev
Apply Changes
Discard Changes
Close


MySQL Table Editor
Table Name: valaszok
Database: tudas
Comment:
Columns and Indices
Table Options
Advanced Options
Column Name
Datatype
NOT NULL
AUTO INC
Flags
Default Value
Comment
nev
CHAR(40)
BINARY
ASCII
fksz
INTEGER
UNSIGNED
ZEROFILL
valasz
CHAR(4)
helyes
Indices
Foreign Keys
Column Details
Index Settings
Index Name:
Index Kind:
INDEX
Index Type:
DEFAULT
Index Columns
(Use Drag'n'Drop)
Apply Changes
Discard Changes
Close

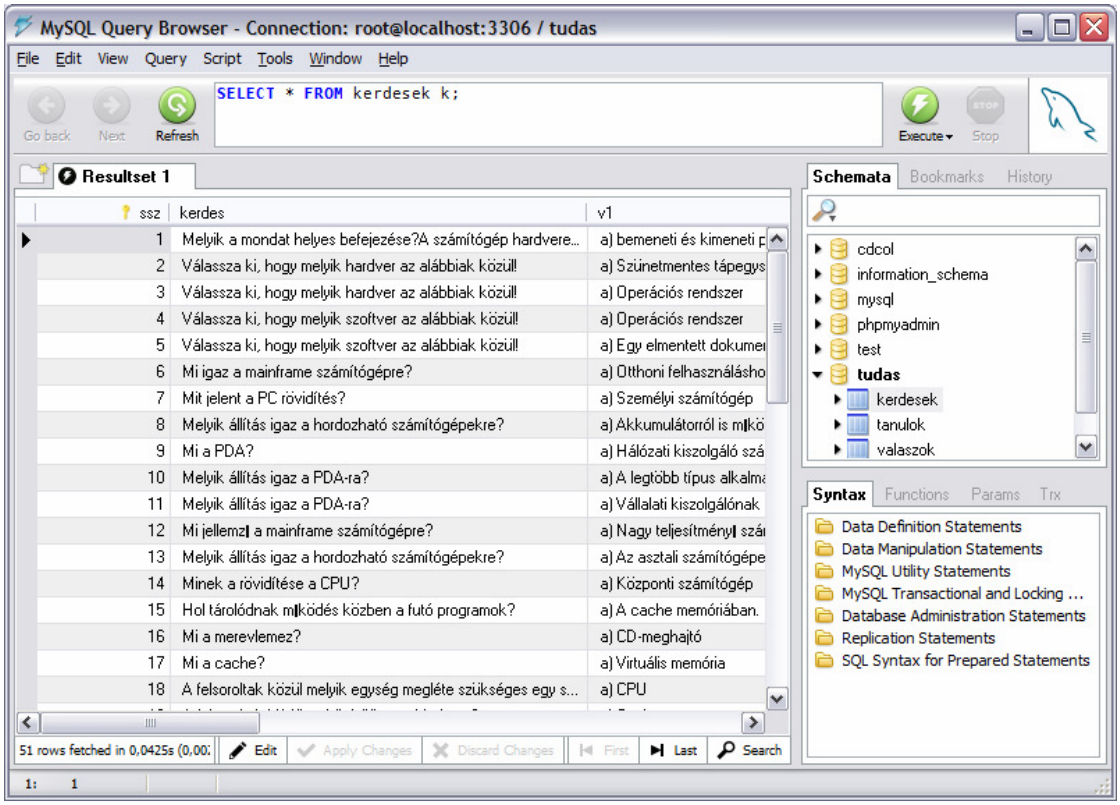
A **MySQL Query Browser** egy grafikus felhasználói felület a MySQL lekérdezésekhez. Legfőbb célja, hogy rugalmas lekérdező és sémakezelő felületet biztosítson. Alapfunkciója SQL lekérdezések végrehajtása, az adatbázisok és táblák kényelmes szerkesztése is lehetővé válik minden lehetőség biztosításával, a létrehozott parancssorok nyomkövetésére is lehetőséget ad számukra. A MySQL Query Browser-ben a felhasználók vizuálisan állíthatják össze a táblákat, azok módosításait, vagy a táblákban tárolt adatokon futtatandó lekérdezéseket. A kliens főként a lekérdezések kialakítását és eredményeinek áttekintését próbálja megkönnyíteni, olyan kényelmi szolgáltatásokkal, mint az eredményekben szabad szöveges keresés, osztott ablakos lekérdezés nézet. A műveletek eredményét a szoftver egy jól áttekinthető, táblázatos formában jeleníti meg, amelyben akár több szűrés eredménye is könnyen összehasonlítható. Tetszőleges számú lekérdezést is futtathatunk, és tarthatunk napirenden. Ezen kívül a program nyilvántartja a lekérdezés történetet, és kedvencek elmentésére is lehetőség nyílik.



A Browser-ben létrehoztam a *kerdesek*, a *tanulok*, és a *valaszok* táblát. Ehhez be kellett állítani a adattáblák szerkezeteit: a mezőneveket, a mező típusait, alapértelmezett értékeit, a mezők hosszát.

SQL parancsokat futtathatunk (pl. lekérdezés: SELECT). A következő képen látható, hogy a *tudas* nevű adatbázis *kerdesek* tábláján duplán kattintva az egérrel, fent

megjelenik a SELECT * FROM kerdesek k; lekérdezés utasítás. Ekkor az Execute gombon kell kattintani a lekérdezés végrehajtásához.

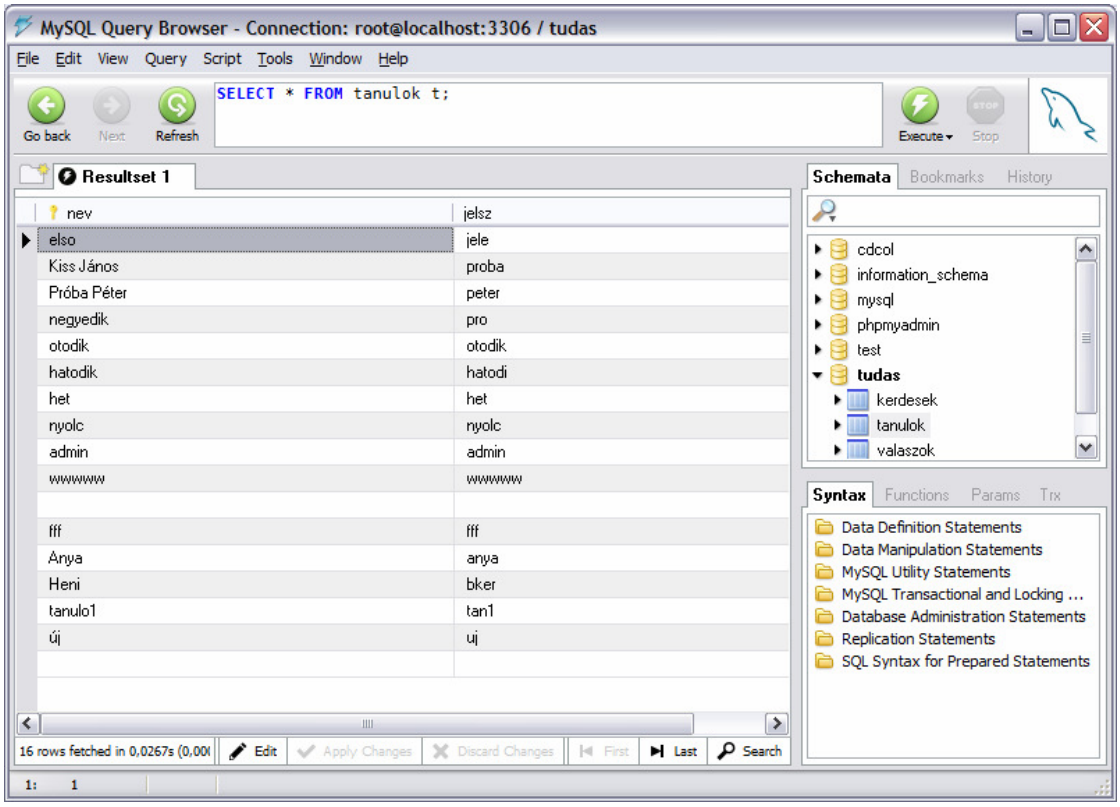


Kevéssé látható, hogy a *kerdesek* tábla 7 oszlopból áll. Az első oszlop a ssz (a kérdések sorszáma), amely elsődleges kulcsú. A második oszlopban találhatók a kérdések. A 3-6 sorban a válaszok vannak, az oszlopok v1-v4-el jelölve. És végül a 7. oszlopban helyezkednek el a helyes válaszok.

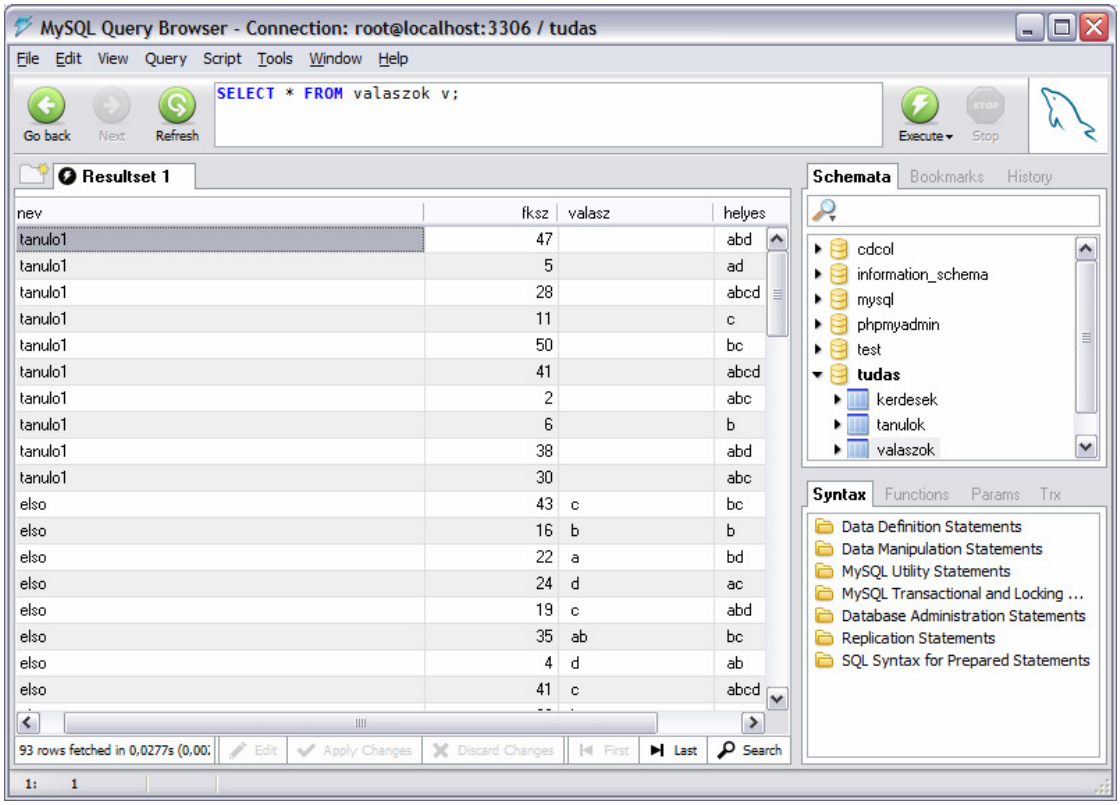
Úgy gondoltam, hogy a kérdések és válaszok megfogalmazásával nem töltök el sok időt, így az az ötletem támadt, hogy az ECDL első moduljának kérdéseivel [12] és válaszaival töltöm fel az adatbázist. Ezek a kérdések a feleletválasztásos (zárt) feladatokhoz tartoznak, azon belül az egyszerű választásos feladatok csoportjába.

A tanulok tábla tartalmát a SELECT * FROM tanulok t parancs segítségével kérhetem le. A *tanulok* táblában két oszlop szerepel. Az első a tanuló neve, a második a jelszava. A név elsődleges kulcsként szerepel.

A panel alsó részén található Edit gombbal lehet szerkeszteni az adattáblák tartalmát. Itt vihetünk be új adatokat, törölhetjük és módosíthatjuk a régieket.



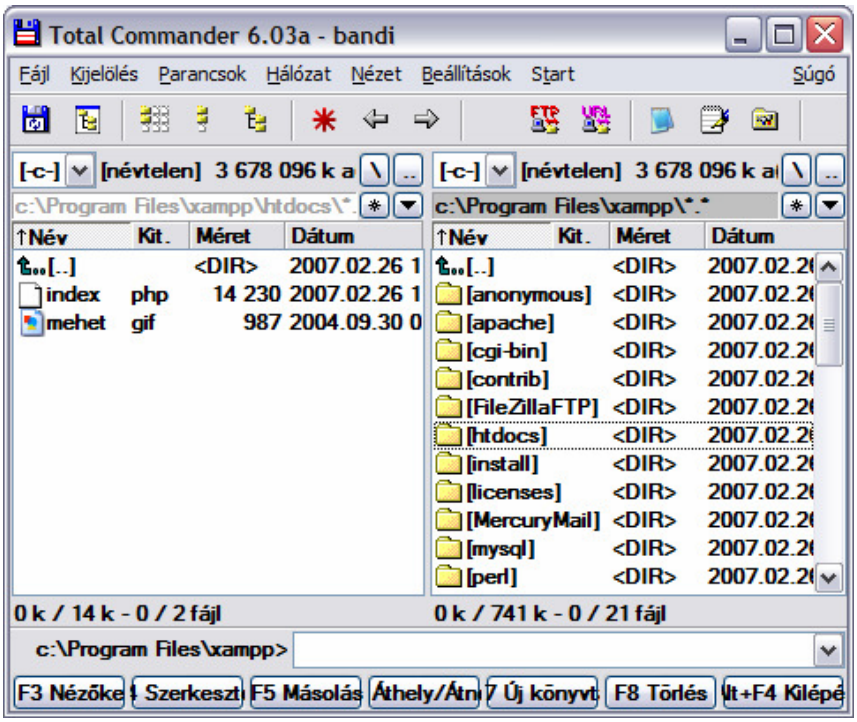
És végül a *valaszok* tábla a megoldott teszt eredményét tárolja. A program az 51 kérdéses adatbázisból generál véletlenszerűen 10 feladatot ismétlődés nélkül.



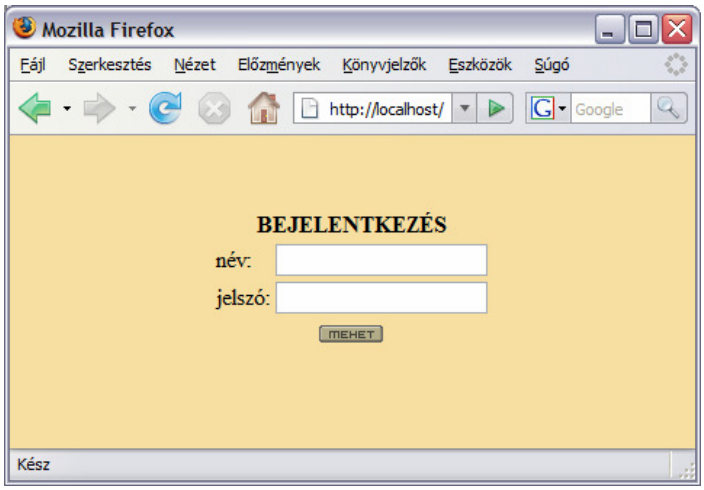
Az első oszlop tartalmazza a feladatlapot megoldó nevét. A második oszlopba az van letárolva, hogy melyik sorszámú feladat lett a tanulónak kiadva, fksz néven (feltett kérdésszám). A harmadik oszlop a tanuló által megadott válaszokat, a negyedik oszlop pedig a helyes válaszok betűjelét tartalmazza.

És most lássuk a HTML kódokat és az eredményét.

Ahhoz, hogy a böngészőbe beírva a localhost nevet rögtön a tesztprogram jelenjen meg, ahhoz először is ki kellett törölni a Program Files\xampp\htdocs mappa tartalmát, és ide kellet létrehozni a PHP-ben megírt szöveges dokumentumot.



Így a következő oldal töltődik be a localhost beírásával:



A programot fel lehet úgy is fogni, mint egy PHP-be ágyazott HTML kódot, de úgy is, mint HTML-be ágyazott PHP kódot.

A HTML utasításokat [4] a szövegben **<** és **>** jelek közé kell zárni. A HTML utasítás kulcsszavaiban nem különböztetjük meg a kisbetűket és a nagybetűket. A HTML dokumentumot két részre lehet bontani a fejlécre és dokumentumtörzsre. A dokumentumot a fejlécelemek vezetik be, melyek kezdetét a **<HEAD>** utasítás jelzi. A fejlécet a **</HEAD>** utasítás zárja. A dokumentumtörzs - amit voltaképpen a WEB-böngésző meg fog jeleníteni - a fájl **<BODY>** és **</BODY>** utasítások közötti része. Ezen elemek között kell elhelyezni mindent: a szöveget, hivatkozásokat, képeket, stb. Amennyiben egy bekezdésen belül mindenképpen új sort szeretnénk kezdeni, akkor a **
** utasítást kell használni.

A dokumentumtörzs a következőképpen néz ki a tesztprogramon belül:

```
<HTML>
<BODY bgcolor="#f6dfa0">                    // az oldal hátterének színe
      .
      .                    // itt található a PHP törzs
      .
</BODY>
</HTML>
```

HTML rész majdhogy csak ennyi található a kódban. Egy PHP utasítás is tartalmazhat HTML kódot, ez az *echo* utasítás. Az *echo* utasításokkal lehet fájlba kiíratni dolgokat, és mivel itt HTML törzsön belül vagyunk, azt a HTML fogja értelmezni. Vagyis a PHP mondja meg a HTML-nek, hogy mit hogyan csináljon.

Ilyen *echo* parancsokkal van megadva a bejelentkezés oldal is:

```
echo('<TABLE width=100% height=100% valign=middle><TR><TD>');
echo('<CENTER><B>BEJELENTKEZÉS</B></CENTER>');
echo("<FORM action='index.php?fel=1' method='post'>
<TABLE align=center><TR><TD><FONT>név:</FONT></TD><TD>");
echo('<INPUT size="20" maxlength="40"
name="c_nameq"><BR></TD></TR><TR><TD><FONT>jelszó:</FONT></TD>
<TD><INPUT TYPE="PASSWORD" size="20" maxlength="6"
name="c_jeleq"><BR></TD>');
echo('</TR></TABLE><CENTER><INPUT type=image
```

```
src="mehet.gif"></CENTER></FORM>');
echo('</TD></TR></TABLE>');
```

A kódrészletben látható a BEJELENTKEZÉS, név, jelszó kiíratása, a név és jelszó utáni mező hossza, a beírható maximum karakterekkel. Látható még a mehet gomb elhelyezése is az oldalon , melyet szintén az xampp\htdocs-ba kellett bemásolni. A mehet gomb gif kiterjesztésű.

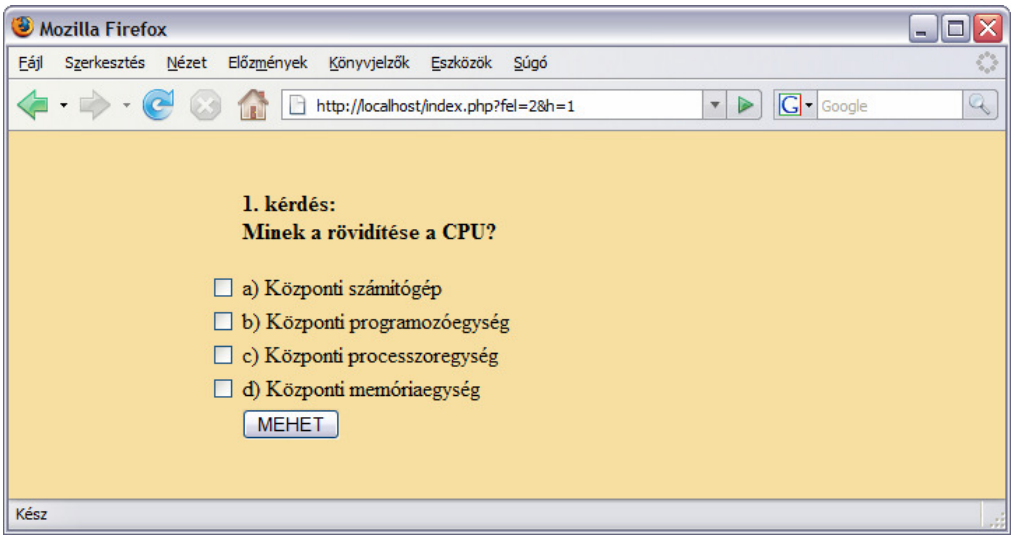
Ha tanulóként lépünk be, akkor vagy az alábbi kiírást fogjuk látni, mely eligazítást ad a teszt megírásához. Felhívja a figyelmet, hogy a teszt csak egyszer végezhető el. Több helyes válasz is létezhet, és csak akkor jár pont, ha az összes helyes választ bejelölik a tanulók.

Az oldal kódja:

```
echo('<CENTER><TABLE width=100% height=100%
valign=middle><TR><TDalign=center>');
echo('<B>Szia '.$nmm.'!</B><BR>');
echo('<BR>Az alábbi oldalakon a tankönyvekből elsajátított tudásodat
mérheted le tesztkérdések segítségével.<BR> A rendszer feltesz neked
10 kérdést, minden kérdéshez 4 válasz tartozik amelyek közül több is
lehet helyes.<BR>');
echo('Jó válasznak az fogadható el, amelyikhez az összes helyes
választ bejelölted! A teszt csak egyszer hajtható végre!<BR>');
echo('Ha indulhatunk nyomd meg a MEHET gombot!<BR><BR>');
echo('<A href=index.php?fel=2&h=1><IMG src=mehet.gif
```

```
border=0></A>');
echo('</TD></TR></TABLE>');
```

A *mehet* gomb megnyomása után indul a teszt.

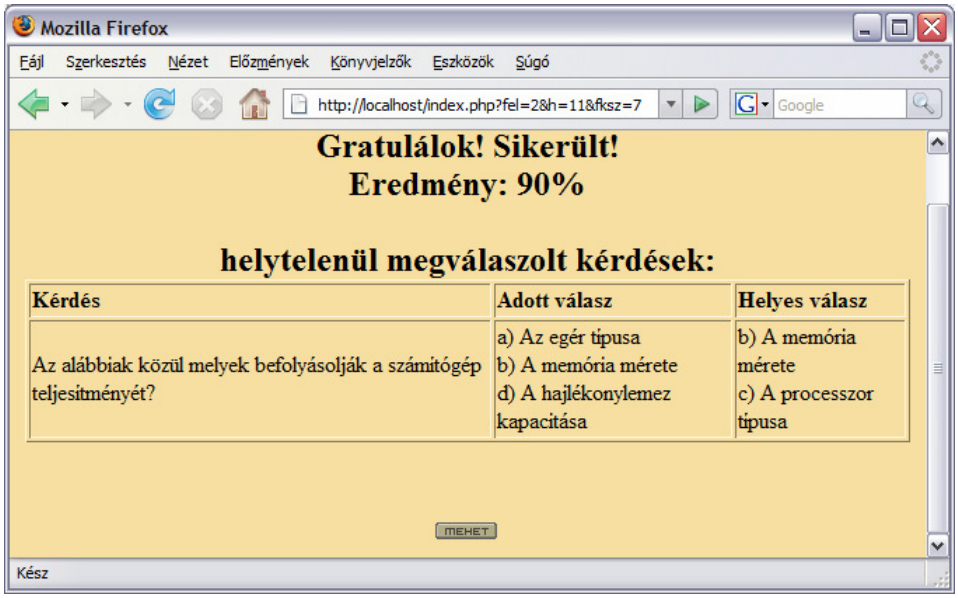


A program a beépített 51 darab kérdésből véletlenszerűen 10 darabot tesz fel, egy kérdést csak egyszer. „Pipával” lehet megjelölni a választ.

```
echo('<TABLE width=100% height=100%><TR><TD align=center><TABLE
width=70% height=70% valign=middle><TR><TD width=10%></TD><TD>');
echo('<FORM action=index.php?fel=2&h='.$kovetkezo.'&fksz='.$be.'
method="post">');                    // megy a válasz
echo('<B>'.$hanyadik.'.
kérdés:<BR>'.$sorika['kerdes'].'</B></TD></TR><TR><TD
align=right><INPUT NAME="val1" TYPE="checkbox">');
echo('</TD><TD>'.$sorika['v1'].'</TD></TR><TR><TD align=right><INPUT
NAME="val2" TYPE="checkbox">');
echo('</TD><TD>'.$sorika['v2'].'</TD></TR><TR><TD align=right><INPUT
NAME="val3" TYPE="checkbox">');
echo('</TD><TD>'.$sorika['v3'].'</TD></TR><TR><TD align=right><INPUT
NAME="val4" TYPE="checkbox">');
echo('</TD><TD>'.$sorika['v4'].'</TD></TR><TR><TD>');
echo('</TD><TD>');
echo('<INPUT TYPE="SUBMIT" VALUE=" MEHET ">');
echo('</TD></TR></TABLE></TD></TR></TABLE>');
echo('</FORM>');
```

Megjelenik a $ jel is, amely kifejezést jelöl a PHP nyelvben. A PHP egy kifejezés-orientált nyelv, abból a szempontból, hogy majdnem minden kifejezés. Minden, aminek értéke van az kifejezés. Tehát az oldalon megjelenik a soron következő kérdés sorszáma, maga a kérdés, és alatta soronként felsorolva a lehetséges válaszok, végül legalul középre rendezve a mehet gomb.

A teszt elvégzése után azonnal lehet látni az eredményt. A teszt 60%-tól sikeres. Az eredmény kiíratása táblázatos módon történik. A táblázat első oszlopában megjelenik az elrontott kérdés, a második oszlopban az adott rossz válasz, és a harmadik oszlopban a helyes válasz. A helytelen válaszok nem abban a sorrendben kerülnek kiírásra, ahogyan fel lettek téve, hanem az adatbázisban elfoglalt sorrendjük szerint



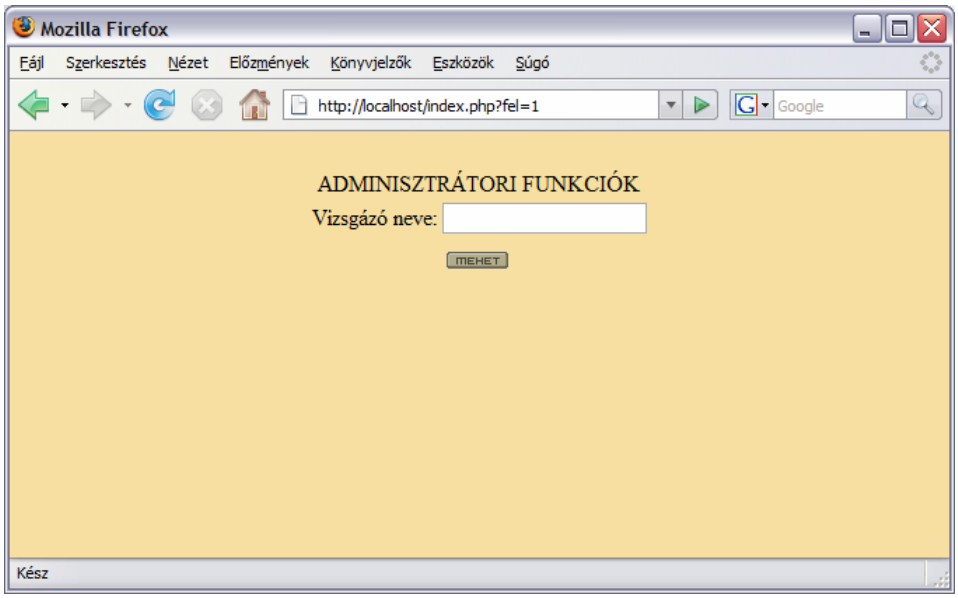

```
echo('<TABLE width=100% height=100%><TR><TD align=center
valign=middle><B><FONT SIZE=+2>');
echo('Gratulálok! Sikerült!');      // ha 60% sikerült, ellenkező esetben:
echo('Sajnos nem sikerült!');
echo('<BR>Eredmény: '.$talalt.'%<BR><BR>helytelenül megválaszolt
kérdések:<BR>');
echo('<TABLE border=1><B><TR><TD><B>Kérdés</B></TD><TD><B>Adott válasz
</B></TD><TD><B>Helyes válasz</B></TD></TR>');
echo('<TR><TD>'.$sorikakerdes['kerdes'].'</TD><TD>'.$valaszst.'</TD><T
D>' .$helyesst.'</TD></TR>');
echo('</TABLE></FONT></TD></TR></TABLE>');
```

A teszt megírása után nem lehet még egyszer megválaszolni a kérdéseket mindaddig, amíg a tanár (admin) ki nem törli az eredményt. Addig a bejelentkezés után az eredmény lap jön be és az oldal tetején egy figyelmeztetés, miszerint a tanuló már végrehajtotta a tesztet.

A figyelmeztetés kódja:

```
echo('<CENTER><FONT SIZE=+2>Te már végrehajtottad a tesztet!
</FONT></CENTER><BR>');
```

Tanárként belépni az *admin* névvel és jeligével lehet. Az alábbi kép jelenik meg bejelentkezés után:



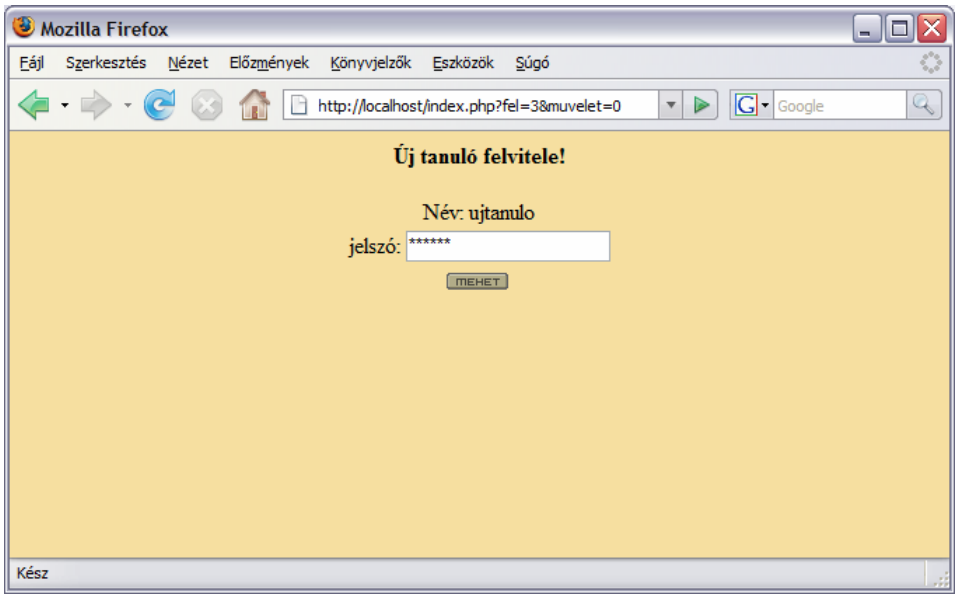
Az oldal kinézetéért felelős kód:

```
echo('<CENTER><BR>ADMINISZTRÁTORI FUNKCIÓK<BR></CENTER>');
echo("<FORM action='index.php?fel=3&muvelet=0' method='post'><TABLE
align=center><TR><TD><FONT>Vizsgázó neve:</FONT></TD><TD>");
echo('<INPUT size="20" maxlength="40"
name="c_nameq"><BR></TD></TR><TR><TD></TD><TD></TD>');
echo('</TR></TABLE><CENTER><INPUT type=image src="mehet.gif">
</CENTER></FORM>');
```

Ha olyan tanuló nevét írjuk be, aki már bent van az adatbázisban, de még nem végezte el a tesztet, akkor a program kilép és újra megjelenik a bejelentkezés oldal. Ha viszont

olyan tanuló nevét írjuk be a vizsgázó neve mezőbe, aki még nem szerepel az adatbázisban, akkor ezzel új tanulót veszünk fel az adatbázisba. Az új tanuló jelszavát is meg kell adnia a tanárnak. A jelszó típusa PASSWORD, ami biztosítja, hogy a titkosítás miatt a megadott jelszó ne betűkkel legyen beírva, hanem csillagokkal.

Az ekkor kapott oldal:



És a hozzá tartozó kódrészlet:

```
echo('<CENTER><B>Új tanuló felvitele!</B><BR><BR>Név: '.$c_nameq);
echo('</CENTER>');
echo("<FORM action='index.php?fel=3&muvelet=2&nevq=".$c_nameq."'
method='post'><TABLE align=center>");
echo('<TR><TD><FONT>jelszó:</FONT></TD><TD><INPUT TYPE="PASSWORD"
size="20" maxlength="6" name="c_jeleq"><BR></TD>');
echo('</TR></TABLE><CENTER><INPUT type=image src="mehet.gif">
</CENTER></FORM>');
```

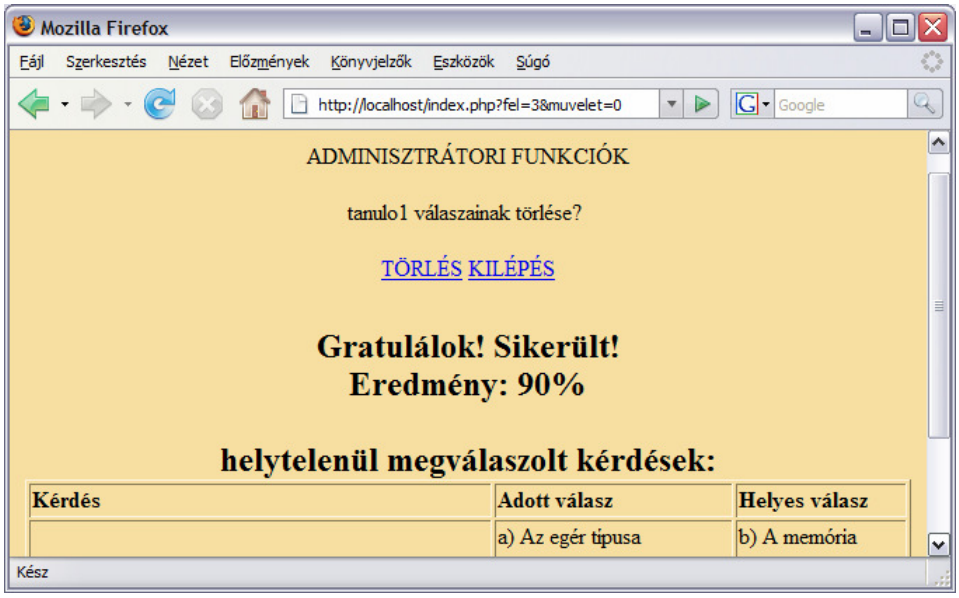
Ezután tájékoztat a program az új tanuló adatbázisba történő sikeres felvételéről.

```
echo('<CENTER><B>Az új adat felvitele megtörtént!</B><BR><A
href=index.php?fel=1><IMG src=mehet.gif border=0></A></CENTER>');
```

Ha viszont olyan tanuló nevét írjuk be a vizsgázó neve mezőbe, aki már kitöltötte a tesztet, akkor megkapjuk a tanuló eredményét. Ugyanaz az eredmény tábla jelenik meg,

mint amit a tanuló is megkapott, de az oldal tetején megjelenik az adminisztrátori funkció kiírás is. A két funkció a törlés és a kilépés. A törléssel törölhetjük a tanuló eredményét. Az eddigiekkel ellentétben itt nem jelenik meg a mehet gomb. Az adatok törlése után kiíródik, hogy az adatok törölve. A mehet gomb megnyomásával ismét a bejelentkezés oldalra jutunk.

```
echo('<CENTER>Az adatok törölve!<BR><BR><A href=index.php><IMG
src=mehet.gif border=0></A></CENTER>');
```




```
echo('<CENTER><BR>ADMINISZTRÁTORI FUNKCIÓK<BR></CENTER>');
echo('<BR><CENTER>'.$c_nameq.' válaszainak törlése? <BR><BR><A
href="index.php?fel=3&muvelet=1&nev='.$c_nameq.'">TÖRLÉS</A>');
echo(' <A href=index.php>KILÉPÉS</A></CENTER>');
```

és ez után az eredmény lekérdezése és kiírása.

Ennyit a HTML-ről és a kiíratásokról. És most lássuk a PHP kódot!

A program írását először a következő kóddal kell kezdeni, ami azt fejezi ki, hogy PHP kódról van szó.

```
<?php
session_start();
if (!isset($visited))
{
   session_register('visited');
```

```
   $visited='';
}
?>
```

Bevezetek az oldalnak egy globális változót, ami a visited (látogató) nevet kapta. Ez után következik a HTML törzs, amiben szintén PHP kód szerepel. PHP-ben függvényeket hozok létre, amik az egész tesztoldal mozgatórugói.

Az első függvény betölti a tanuló vagy akár az admin nevét és jelszavát.

```
function load_tanulo($be,$jel) {
      global $nmm;        // globális változó
      $nmm='';
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM tanulok WHERE nev='".$be."'";
      $eredmeny = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmeny)>0) {
         while ($sorika = mysql_fetch_assoc($eredmeny)) {
            $_name=$sorika['nev'];
            $_jelsz=$sorika['jelsz']; }
         $log=0;
         if ((strtoupper($_name)==strtoupper($be)) and
($_jelsz==$jel)){
            $nmm=$_name; } };
      mysql_free_result($eredmeny);
      mysql_close($kapcsolat);}
```

A mysql_connect kapcsolatot nyit meg egy MySQL szerverhez, ha a kapcsolódás sikertelen, akkor a mysql_error az előző MySQL művelet hibaszövegét adja. A kapcsolat a szerverrel a lehető leghamarabb bezárul, amikor a szkript befejezi működését, kivéve ha korábban be lett zárva explicit módon a mysql_close() meghívásával. A mysql_select_db kiválaszt egy MySQL adatbázist, itt a tudas adatbázist. Végrehajt egy SQL lekérdezést, ami az adatbázisban szereplő nevek között megnézi, hogy szerepel-e a név mezőbe beírt név. A mysql_query MySQL kérést küld a szervernek A mysql_num_rows az eredményben szereplő sorok számát adja vissza. A

függvény csak SELECT kérésre használható. A mysql_fetch_assoc függvény az eredmény egy sorát asszociatív tömbként adja vissza. Vagyis ha a beírt nevet megtalálja az adatbázisban, akkor berakja egy tömbbe. A while ciklusban fel kell bontani a kapott eredményt névre és jelszóra. A mysql_free_result felszabadítja az eredmény által lefoglalt memóriát.

A második függvény az aktuális látogató a megfelelő kérdésekre adott válaszát tárolja el a kerdesek táblában.

```
function valasz_kiir($fkszx,$adottx,$visitedx) {
      $kapcsolat = mysql_connect('localhost', 'root', '') or die('Nem
tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM kerdesek WHERE ssz=".$fkszx;
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      while ($sorika = mysql_fetch_assoc($eredmenyx)) {
         $_jo=$sorika['jo']; }
      $keres = "INSERT INTO valaszok VALUES
('".$visitedx."',".$fkszx.",'".$adottx."','".$_jo."')";

      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      mysql_close($kapcsolat); }
```

A kerdesek adatbázisban megkeresem a feltett kérdés sorát. A kapott eredményt berakom az eredmény tömbbe. A sorból kiveszem a jó megoldást és értékül adom a `$_jo` kifejezésnek. Az `INSERT INTO...` SQL paranccsal új rekordot illesztek be a valaszok táblába. A név oszlopot a tanuló nevével töltöm ki, ami a `$visitedx` kifejezés, a feltett kérdésszám (`$fkszx`) a második oszlopba kerül, a válasz oszlopba a `$adottx` kifejezés értéke, és végül a helyes oszlopba a `$_jo` kifejezés értéke tárolódik le.

A harmadik PHP függvény generál egy számot egytől 51-ig, és az így kapott sorszámú kérdést teszi fel a vizsgázónak. (`$fkszx=mt_rand(1, 51); )` A függvény ellenőrzi azt is, nem tette-e már fel ugyanazt a kérdést. Ha a valaszok táblában eltárolt kérdések

sorszáma megegyezik egy már feltettével, akkor másik kérdés generálódik. A ciklus elöltesztelő, mivel először vizsgálni kell és csak utána teheti fel a kérdést. A kérdés feltevése után hiába vizsgálná.

```
function letezo($visitedx,$fkszx) {
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      do {
      $keres = "SELECT * FROM valaszok WHERE nev='".$visitedx."' and
fksz=".$fkszx;
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmenyx)<>0) {
         $fkszx=mt_rand(1, 51); } }
      while (mysql_num_rows($eredmenyx)<>0);
      mysql_close($kapcsolat);
      return $fkszx;  }
```

A következő függvény végzi az adott válaszok kiértékelését. Ha a jó válaszok száma eléri a 6-ot, akkor a vizsgát sikeresnek veszi. Összehasonlítja az adott válaszokat a helyes válaszokkal, és ha teljes az egyezés a két sztring között, akkor ad rá pontot. Összekapcsolja az a választ a v1 oszloppal, a b választ a v2 oszloppal, a c-t a v3-mal, és a d választ a v4 oszloppal. Így ha van rossz megoldás, akkor mind az adott válasz oszlopába, mind pedig a helyes válasz oszlopába az adott kérdéshez tartozó betűjelen kívül kiírja magát a választ is. Az eredményt úgy írja ki, hogy a jó válaszok száma + 0% (pl.:jó válasz:9 -> 90%), kivéve, ha egy jó válasz sem volt, mert akkor 0%.

```
function kiertekel($visitedx) {
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM valaszok WHERE valasz=helyes and
nev='".$visitedx."'";
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
```

```
      $talalt=mysql_num_rows($eredmenyx);
     echo...
      if ($talalt>=6) { echo... }          //sikerült
      else { echo...                 //nem sikerült
      if ($talalt==0){
 echo...          //eredmény:$talalt, helytelenül megválaszolt kérdések }
 else {
 echo...          //helytelenül megválaszolt kérdések
 }
      $keres = "SELECT * FROM valaszok WHERE valasz<>helyes and
nev='".$visitedx."'";
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      $sorx=1;
      echo...              // kérdés, adott válasz, helyes válasz
      while ($sorika = mysql_fetch_assoc($eredmenyx))
      {
         $keres2 = "SELECT * FROM kerdesek WHERE
ssz=".$sorika['fksz'];
         $eredmeny2 = mysql_query($keres2) or die('Hiba a kérésben: '
. mysql_error());
         $sorikakerdes=mysql_fetch_assoc($eredmeny2);
         $valaszst='';
         $helyesst='';
         if (strpos($sorika['valasz'],'a')!== false)
{$valaszst=$sorikakerdes['v1'].'<BR>';}
         if (strpos($sorika['valasz'],'b')!== false)
{$valaszst=$valaszst.$sorikakerdes['v2'].'<BR>';}
         if (strpos($sorika['valasz'],'c')!== false)
{$valaszst=$valaszst.$sorikakerdes['v3'].'<BR>';}
         if (strpos($sorika['valasz'],'d')!== false)
{$valaszst=$valaszst.$sorikakerdes['v4'].'<BR>';}
         if (strpos($sorika['helyes'],'a')!== false)
{$helyesst=$sorikakerdes['v1'].'<BR>';}
         if (strpos($sorika['helyes'],'b')!== false)
{$helyesst=$helyesst.$sorikakerdes['v2'].'<BR>';}
         if (strpos($sorika['helyes'],'c')!== false)
{$helyesst=$helyesst.$sorikakerdes['v3'].'<BR>';}
```

```
         if (strpos($sorika['helyes'],'d')!== false)
{$helyesst=$helyesst.$sorikakerdes['v4'].'<BR>';}
        echo...
 $sorx=$sorx+1;}
      echo...
      mysql_close($kapcsolat); }
```

Az alábbi függvénynek csupán az a feladata, hogy feltegye a kérdést a tanuló számára:

```
function load_kerdes($be,$hanyadik) {
      echo...
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $kovetkezo=$hanyadik+1;
      echo...              // fel=2 megy a válasz
      $keres = "SELECT * FROM kerdesek WHERE ssz='".$be."'";
      $eredmeny = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmeny)>0) {
        while ($sorika = mysql_fetch_assoc($eredmeny)) {
            echo...          // a kérdés és a négy alternatíva
         }};
     echo...
     mysql_free_result($eredmeny);
     mysql_close($kapcsolat);}
```

És végül a vezérlést megvalósító belép függvény, ami elvégzi a megfelelő műveleteket. Eldönti, hogy tanuló vagy az oktató lép be a rendszerbe. Végrehajtja a belépést, új tanuló felvételét az adatbázisba, a meglévő eredmények lekérdezését, törlését. Lebonyolítja a vizsgáztatást, elvégzi a kiértékelést és az eredmények kiíratását. A függvény helyenként meghívja a többi függvényt.

```
function bejel() {
   global $nmm;
   echo...        // bejelentkezés }
   if ($fel==1) {
      load_tanulo($c_nameq,$c_jeleq);
```

```
      if ($nmm<>''){
         if ($nmm=='admin') {
            $visited=$nmm;
            echo... }          // adminisztrátori funkciók : vizsgázó neve
         else {
            $marvizsgazott=0;
            $kapcsolat = mysql_connect('localhost', 'root', '')
            or die('Nem tudok csatlakozni: ' . mysql_error());
            mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
            $keres = "SELECT * FROM valaszok WHERE nev='".$nmm."'";
            $eredmeny = mysql_query($keres) or die('Hiba a kérésben: '
. mysql_error());
            if (mysql_num_rows($eredmeny)>0) {
               $marvizsgazott=1; }
            mysql_free_result($eredmeny);
            mysql_close($kapcsolat);
            if ($marvizsgazott==0) {
               $visited=$nmm;
               echo... }              //információk a teszt írásához
            else {
                echo...          // már végrehajtotta a tesztet
               kiertekel($nmm); }}}
      else {
         bejel();}}
   else {
      if ($fel==2) {
         if ($visited<>'') {
    if ($h>1) {
               $adott='';
               if ($val1=='on') {$adott='a';}
               if ($val2=='on') {$adott=$adott.'b';}
        if ($val3=='on') {$adott=$adott.'c';}
               if ($val4=='on') {$adott=$adott.'d';}
               valasz_kiir($fksz,$adott,$visited); }
            $veletlenkerdes=letezo($visited,mt_rand(1, 51));
            if ($h<11) {load_kerdes($veletlenkerdes,$h);}
            else {
               kiertekel($visited);
```

```
                $visited='';
                echo... } } }
      else {
         if (($fel==3) and ($visited=='admin')) {
            if ($muvelet==0) {
               $kapcsolatadm2 = mysql_connect('localhost', 'root',
'')
               or die('Nem tudok csatlakozni: ' . mysql_error());
               mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
               $keres = "SELECT * FROM tanulok WHERE
nev='".$c_nameq."'";
               $eredmenyxx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
               $talalt=mysql_num_rows($eredmenyxx);
               mysql_close($kapcsolatadm2);
               if ($talalt>0) {
                  $talalt=0;
                  $kapcsolatadm = mysql_connect('localhost', 'root',
'')
                  or die('Nem tudok csatlakozni: ' . mysql_error());
                  mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
                  $keres = "SELECT * FROM valaszok WHERE
nev='".$c_nameq."'";
                  $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
                  $talalt=mysql_num_rows($eredmenyx);
                  mysql_close($kapcsolatadm);
                  if ($talalt>0) {
                     echo...          // eredmény törlése vagy kilépés
                     kiertekel($c_nameq); }
                  else {
                     bejel(); } }
               else {
                  echo... } }       // új tanulófelvitele
            if ($muvelet==1) {
               echo...              //adminisztrátori funkciók
               $kapcsolat = mysql_connect('localhost', 'root', '')
```

```
                or die('Nem tudok csatlakozni: ' . mysql_error());
                mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
                $keres = "DELETE FROM valaszok WHERE nev='".$nev."'";
                $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
                mysql_close($kapcsolat);
                echo... }              //adatok törölve
            if ($muvelet==2) {
                $kapcsolat = mysql_connect('localhost', 'root', '')
                or die('Nem tudok csatlakozni: ' . mysql_error());
                mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
                $keres = "INSERT INTO tanulok VALUES
('".$nevq."','".$c_jeleq."')";
                $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
                mysql_close($kapcsolat);
                echo... } }            //az új adat felvitele sikeres
         else {
            bejel();} } }
```

6. Összefoglalás

A számítógép nagyon megkönnyítheti a tanárok munkáját is. Annak az oka, hogy még mindig kevesen használják a számítógépet munkájuk során az, hogy nem rendelkeznek kellő ismerettel a számítógépek kezelésével, használatával kapcsolatban. Meg kellene próbálnunk lépést tartani a fejlődéssel, mert a fejlett eszközök használatával a mai rohanó világunkban oly fontos dolgot takaríthatunk meg, mint az idő. Az ezáltal nyert szabadidőt felhasználhatjuk számunkra hasznosabb dolgokra.

Persze vannak az oktatásnak olyan területei, ahol a számítógép sosem tudná pótolni a tanárt, de nagyon sok területen segítségére lehet tanárnak és diáknak egyaránt. Például a számítógép tesztekkel történő vizsgáztatásra kiválóan alkalmas, mert megbízható, pontos, gyors és hatékony.

A cél az, hogy az iskolákban az oktatás segítése, a tanítás jobb szervezése, hatékonyabbá tétele érdekében minél szélesebb körben használják a számítógépek révén szerzett előnyöket.

Köszönetnyilvánítás:

Köszönet dr. Nyakóné dr. Juhász Katalin tanárnőnek segítőkészségéért, kedvességéért és szellemi támogatásáért.

Külön köszönet illeti dr. Papp Zoltán tanár urat türelméért, és amiért lehetővé tette számomra a tanári szak felvételét.

Irodalomjegyzék

1. Bódy Bence: Az SQL példákon keresztül
   Jedlik Oktatási Stúdió, Budapest, 2004
2. Bódi Zoltán: A világháló nyelve
   Kiskapu Kiadó, Budapest, 2004
3. Falus Iván: Didaktika – Elméleti alapok a tanítás tanulásához
   Nemzeti Tankönyvkiadó, Budapest, 2003
4. Gál Tibor: Web programozás
   Műegyetem Kiadó, Budapest, 2004
5. Julie C. Meloni: A PHP, a MySQL és az Apache használata
   Panem Kiadó, Budapest, 2004
6. Matt Zadanstra: Tanuljuk meg a PHP5 használatát 24 óra alatt
   Kiskapu Kiadó, Budapest, 2004
7. Nyakóné Juhász Katalin: Az informatika iskolai alkalmazásai
   Jegyzet, Debreceni Egyetem
8. Pluhár Emese: Internet kisszótár
   Kiskapu Kiadó, Budapest, 2004
9. Rajtik János: Alapismeretek – Hálózatok, Várbíróné Nahaji Anikó: Internet
   Pedellus Tankönyvkiadó Kft., Debrecen, 2000
10. Síkos László: Szerver oldali webprogramozás
    Kiskapu Kiadó, Budapest, 2004
11. Stolnicki Gyula: SQL kézikönyv
    ComputerBooks, Budapest, 1998
12. Váradi Zsolt: Vizsgapéldatár – Az Európai Számítógép-használói Jogosítvány vizsgafeladatai
    Kossuth Kiadó, Budapest, 2003

Függelék

```
<?php
session_start();
if (!isset($visited))
{
   session_register('visited'); //oldal globalis valt
   $visited='';
}
?>
<HTML>
<BODY bgcolor="#f6dfa0">
<?php
   function load_tanulo($be,$jel)
   {
      global $nmm;
      $nmm='';
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM tanulok WHERE nev='".$be."'";
      $eredmeny = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmeny)>0)
      {
         while ($sorika = mysql_fetch_assoc($eredmeny))
         {
            $_name=$sorika['nev'];
            $_jelsz=$sorika['jelsz'];
         }
         $log=0;
         if ((strtoupper($_name)==strtoupper($be)) and
($_jelsz==$jel))
         {
            $nmm=$_name;
         }
      };
      mysql_free_result($eredmeny);
      mysql_close($kapcsolat);
   }

   function valasz_kiir($fkszx,$adottx,$visitedx)
   {
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM kerdesek WHERE ssz=".$fkszx;
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      while ($sorika = mysql_fetch_assoc($eredmenyx))
      {
         $_jo=$sorika['jo'];
      }
      $keres = "INSERT INTO valaszok VALUES
('".$visitedx."',".$fkszx.",'".$adottx."','".$_jo."')";
```

```
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      mysql_close($kapcsolat);
   }

   function letezo($visitedx,$fkszx)
   {
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      do
      {
      $keres = "SELECT * FROM valaszok WHERE nev='".$visitedx."' and
fksz=".$fkszx;
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmenyx)<>0)
      {
         $fkszx=mt_rand(1, 51);
      }
      }
      while (mysql_num_rows($eredmenyx)<>0);
      mysql_close($kapcsolat);
      return $fkszx;
   }

   function kiertekel($visitedx)
   {
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $keres = "SELECT * FROM valaszok WHERE valasz=helyes and
nev='".$visitedx."'";
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      $talalt=mysql_num_rows($eredmenyx);
      echo('<TABLE width=100% height=100%><TR><TD align=center
valign=middle><B><FONT SIZE=+2>');
      if ($talalt>=6) {echo('Gratulálok! Sikerült!');}
      else {echo('Sajnos nem sikerült!');}
      if ($talalt==0){
 echo('<BR>Eredmény: '.$talalt.'%<BR><BR>helytelenül megválaszolt
kérdések:<BR>');
 }
 else {
 echo('<BR>Eredmény: '.$talalt.'0%<BR><BR>helytelenül megválaszolt
kérdések:<BR>');
 }
      $keres = "SELECT * FROM valaszok WHERE valasz<>helyes and
nev='".$visitedx."'";
      $eredmenyx = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      $sorx=1;
      echo('<TABLE border=1><B><TR><TD><B>Kérdés</B></TD><TD><B>Adott
válasz</B></TD><TD><B>Helyes válasz</B></TD></TR>');
      while ($sorika = mysql_fetch_assoc($eredmenyx))
```

```
      {
         $keres2 = "SELECT * FROM kerdesek WHERE
ssz=".$sorika['fksz'];
         $eredmeny2 = mysql_query($keres2) or die('Hiba a kérésben: '
. mysql_error());
         $sorikakerdes=mysql_fetch_assoc($eredmeny2);
         $valaszst='';
         $helyesst='';
         if (strpos($sorika['valasz'],'a')!== false)
{$valaszst=$sorikakerdes['v1'].'<BR>';}
         if (strpos($sorika['valasz'],'b')!== false)
{$valaszst=$valaszst.$sorikakerdes['v2'].'<BR>';}
         if (strpos($sorika['valasz'],'c')!== false)
{$valaszst=$valaszst.$sorikakerdes['v3'].'<BR>';}
         if (strpos($sorika['valasz'],'d')!== false)
{$valaszst=$valaszst.$sorikakerdes['v4'].'<BR>';}
         if (strpos($sorika['helyes'],'a')!== false)
{$helyesst=$sorikakerdes['v1'].'<BR>';}
         if (strpos($sorika['helyes'],'b')!== false)
{$helyesst=$helyesst.$sorikakerdes['v2'].'<BR>';}
         if (strpos($sorika['helyes'],'c')!== false)
{$helyesst=$helyesst.$sorikakerdes['v3'].'<BR>';}
         if (strpos($sorika['helyes'],'d')!== false)
{$helyesst=$helyesst.$sorikakerdes['v4'].'<BR>';}

echo('<TR><TD>'.$sorikakerdes['kerdes'].'</TD><TD>'.$valaszst.'</TD><T
D>'.$helyesst.'</TD></TR>');
         $sorx=$sorx+1;
      }
      echo('</TABLE></FONT></TD></TR></TABLE>');
      mysql_close($kapcsolat);
   }

   function load_kerdes($be,$hanyadik)
   {
      echo('<TABLE width=100% height=100%><TR><TD align=center><TABLE
width=70% height=70% valign=middle><TR><TD width=10%></TD><TD>');
      $kapcsolat = mysql_connect('localhost', 'root', '')
      or die('Nem tudok csatlakozni: ' . mysql_error());
      mysql_select_db('tudas') or die('Nem sikerült kiválasztanom az
adatbázist');
      $kovetkezo=$hanyadik+1;
      echo('<FORM action=index.php?fel=2&h='.$kovetkezo.'&fksz='.$be.'
method="post">'); //fel=2 megy a válasz
      $keres = "SELECT * FROM kerdesek WHERE ssz='".$be."'";
      $eredmeny = mysql_query($keres) or die('Hiba a kérésben: ' .
mysql_error());
      if (mysql_num_rows($eredmeny)>0)
      {
        while ($sorika = mysql_fetch_assoc($eredmeny))
         {
            echo('<B>'.$hanyadik.'.
kérdés:<BR>'.$sorika['kerdes'].'</B></TD></TR><TR><TD
align=right><INPUT NAME="val1" TYPE="checkbox">');
            echo('</TD><TD>'.$sorika['v1'].'</TD></TR><TR><TD
align=right><INPUT NAME="val2" TYPE="checkbox">');
            echo('</TD><TD>'.$sorika['v2'].'</TD></TR><TR><TD
align=right><INPUT NAME="val3" TYPE="checkbox">');
            echo('</TD><TD>'.$sorika['v3'].'</TD></TR><TR><TD
```

```
align=right><INPUT NAME="val4" TYPE="checkbox">');
            echo('</TD><TD>'.$sorika['v4'].'</TD></TR><TR><TD>');
         }
      };
     echo('</TD><TD>');
     echo('<INPUT TYPE="SUBMIT" VALUE=" MEHET ">');
     echo('</TD></TR></TABLE></TD></TR></TABLE>');
     echo('</FORM>');
     mysql_free_result($eredmeny);
     mysql_close($kapcsolat);

   }

   function bejel()
   {
   global $nmm;
   echo('<TABLE width=100% height=100% valign=middle><TR><TD>');
      echo('<CENTER><B>BEJELENTKEZÉS</B></CENTER>');
      echo("<FORM action='index.php?fel=1' method='post'><TABLE
align=center><TR><TD><FONT>név:</FONT></TD><TD>");
      echo('<INPUT size="20" maxlength="40"
name="c_nameq"><BR></TD></TR><TR><TD><FONT>jelszó:</FONT></TD><TD><INP
UT TYPE="PASSWORD" size="20" maxlength="6"
name="c_jeleq"><BR></TD>');
      echo('</TR></TABLE><CENTER><INPUT type=image
src="mehet.gif"></CENTER></FORM>');
   echo('</TD></TR></TABLE>');
   }

   if ($fel==1)
   {
      load_tanulo($c_nameq,$c_jeleq);
      if ($nmm<>'')
      {
         if ($nmm=='admin')
         {
            $visited=$nmm;
            echo('<CENTER><BR>ADMINISZTRÁTORI FUNKCIÓK<BR></CENTER>');

            echo("<FORM action='index.php?fel=3&muvelet=0'
method='post'><TABLE align=center><TR><TD><FONT>Vizsgázó
neve:</FONT></TD><TD>");
            echo('<INPUT size="20" maxlength="40"
name="c_nameq"><BR></TD></TR><TR><TD></TD><TD></TD>');
            echo('</TR></TABLE><CENTER><INPUT type=image
src="mehet.gif"></CENTER></FORM>');
         }
         else
         {
            $marvizsgazott=0;
            $kapcsolat = mysql_connect('localhost', 'root', '')
            or die('Nem tudok csatlakozni: ' . mysql_error());
            mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
            $keres = "SELECT * FROM valaszok WHERE nev='".$nmm."'";
            $eredmeny = mysql_query($keres) or die('Hiba a kérésben: '
. mysql_error());
            if (mysql_num_rows($eredmeny)>0)
            {
```

```
                $marvizsgazott=1;
            }
            mysql_free_result($eredmeny);
            mysql_close($kapcsolat);
            if ($marvizsgazott==0)
            {
                $visited=$nmm;
                echo('<CENTER><TABLE width=100% height=100%
valign=middle><TR><TD align=center>');
                echo('<B>Szia '.$nmm.'!</B><BR>');
                echo('<BR>Az alábbi oldalakon a tankönyvekből
elsajátított tudásodat mérheted le tesztkérdések segítségével.<BR> A
rendszer feltesz neked 10 kérdést, minden kérdéshez 4 válasz tartozik
amelyek közül több is lehet helyes.<BR>');
                echo('Jó válasznak az fogadható el, amelyikhez az
összes helyes választ bejelölted! A teszt csak egyszer hajtható
végre!<BR>');
                echo('Ha indulhatunk nyomd meg a MEHET
gombot!<BR><BR>');
                echo('<A href=index.php?fel=2&h=1><IMG src=mehet.gif
border=0></A>');
                echo('</TD></TR></TABLE>');
            }
            else
            {
                 echo('<CENTER><FONT SIZE=+2>Te már végrehajtottad a
tesztet!</FONT></CENTER><BR>');
                 echo('</FONT><BR><CENTER><A href=index.php><IMG
src=mehet.gif border=0></CENTER>');
                 kiertekel($nmm);
            }
         }
      }
      else
      {
         bejel();
      }
   }
   else
   {
      if ($fel==2)
      {
         if ($visited<>'')
         {
    if ($h>1)
            {
               $adott='';
               if ($val1=='on') {$adott='a';}
               if ($val2=='on') {$adott=$adott.'b';}
        if ($val3=='on') {$adott=$adott.'c';}
               if ($val4=='on') {$adott=$adott.'d';}
               valasz_kiir($fksz,$adott,$visited);
            }
            $veletlenkerdes=letezo($visited,mt_rand(1, 51));
            if ($h<11) {load_kerdes($veletlenkerdes,$h);}
            else
            {
               kiertekel($visited);
               $visited='';
```

```
                echo('<BR><CENTER><A href=index.php><IMG src=mehet.gif
border=0></A></CENTER>');
             }
          }
       }
       else
       {
          if (($fel==3) and ($visited=='admin'))
          {
             if ($muvelet==0)
             {
                $kapcsolatadm2 = mysql_connect('localhost', 'root',
'')
                or die('Nem tudok csatlakozni: ' . mysql_error());
                mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
                $keres = "SELECT * FROM tanulok WHERE
nev='".$c_nameq."'";
                $eredmenyxx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
                $talalt=mysql_num_rows($eredmenyxx);
                mysql_close($kapcsolatadm2);
                if ($talalt>0)
                {
                   $talalt=0;
                   $kapcsolatadm = mysql_connect('localhost', 'root',
'')
                   or die('Nem tudok csatlakozni: ' . mysql_error());
                   mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
                   $keres = "SELECT * FROM valaszok WHERE
nev='".$c_nameq."'";
                   $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
                   $talalt=mysql_num_rows($eredmenyx);
                   mysql_close($kapcsolatadm);
                   if ($talalt>0)
                   {
                      echo('<CENTER><BR>ADMINISZTRÁTORI
FUNKCIÓK<BR></CENTER>');
                      echo('<BR><CENTER>'.$c_nameq.' válaszainak
törlése? <BR><BR><A
href="index.php?fel=3&muvelet=1&nev='.$c_nameq.'">TÖRLÉS</A>');
                      echo(' <A href=index.php>KILÉPÉS</A></CENTER>');
                      kiertekel($c_nameq);
                   }
                   else
                   {
                      bejel();
                   }
                }
                else
                {
                   echo('<CENTER><B>Új tanuló
felvitele!</B><BR><BR>Név: '.$c_nameq);
                   echo('</CENTER>');
                   echo("<FORM
action='index.php?fel=3&muvelet=2&nevq=".$c_nameq."'
method='post'><TABLE align=center>");
```

```
                  echo('<TR><TD><FONT>jelszó:</FONT></TD><TD><INPUT
TYPE="PASSWORD" size="20" maxlength="6" name="c_jeleq"><BR></TD>');
                  echo('</TR></TABLE><CENTER><INPUT type=image
src="mehet.gif"></CENTER></FORM>');
               }
            }
            if ($muvelet==1)
            {
               echo('<CENTER><BR>ADMINISZTRÁTORI
FUNKCIÓK<BR></CENTER>');
               $kapcsolat = mysql_connect('localhost', 'root', '')
               or die('Nem tudok csatlakozni: ' . mysql_error());
               mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
               $keres = "DELETE FROM valaszok WHERE nev='".$nev."'";
               $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
               mysql_close($kapcsolat);
               echo('<CENTER>Az adatok törölve!<BR><BR><A
href=index.php><IMG src=mehet.gif border=0></A></CENTER>');
            }
            if ($muvelet==2)
            {
               $kapcsolat = mysql_connect('localhost', 'root', '')
               or die('Nem tudok csatlakozni: ' . mysql_error());
               mysql_select_db('tudas') or die('Nem sikerült
kiválasztanom az adatbázist');
               $keres = "INSERT INTO tanulok VALUES
('".$nevq."','".$c_jeleq."')";
               $eredmenyx = mysql_query($keres) or die('Hiba a
kérésben: ' . mysql_error());
               mysql_close($kapcsolat);
               echo('<CENTER><B>Az új adat felvitele
megtörtént!</B><BR><A href=index.php?fel=1><IMG src=mehet.gif
border=0></A></CENTER>');
            }
         }
         else
         {
            bejel();
         }
      }
   }
?>
</BODY>
</HTML>
```