Univerzita Karlova v Praze
Pedagogická fakulta

DISERTAČNÍ PRÁCE

2011 Mikoláš Panský

Univerzita Karlova v Praze
Pedagogická fakulta

Rozvoj databázového myšlení v kontextu školního vzdělávání

Mikoláš Panský

2011

PROHLÁŠENÍ

Prohlašuji, že jsem disertační práci na téma Rozvoj databázového myšlení v kontextu školního vzdělávání vypracoval na základě vlastních zjištění a za použití v práci uvedených pramenů a literatury. Dále prohlašuji, že tato disertační práce nebyla využita k získání jiného nebo stejného titulu.

V Praze dne 30. dubna 2011

.....

Mikoláš Panský

Rád bych touto cestou vyjádřil poděkování doc. PaedDr. Radce Wildové, CSc., doc. RNDr. Miroslavě Černochové, CSc., a Ing. Jaroslavu Novákovi, PhD., za odborné vedení a cenné rady při psaní disertační práce. Zároveň děkuji své rodině za zázemí a podporu.

NÁZEV:
Rozvoj databázového myšlení v kontextu školního vzdělávání

AUTOR:
Mikoláš Panský

ABSTRAKT:
Hlavním tématem práce je databázové myšlení. Jedná se o soubor kompetencí a způsobu myšlení vedoucího k osvojení práce s databázovým systémem. Rozvoj databázového myšlení je pedagogické působení na žáka za účelem formování odpovídající kognitivní struktury a nácviku praktických dovedností. Teoretická část se skládá z literární rešerše cílů, obsahu a výukových metod. Praktická část je orientována na metodiky a evaluační kritéria. Výstupem práce je příspěvek didaktice ICT zdůrazněním významu výuky databází.

KLÍČOVÁ SLOVA:
pedagogika, databáze, databázové myšlení, strukturovaný dotazovací jazyk

TITLE:
Database thinking development in Context of School Education

AUTHOR:
Mikoláš Panský

ABSTRACT:
The term database thinking is understood as group of Competencies that enables working with Database System. Database thinking development is targeted educational incidence to student with the expected outcome ability working with Database system. Thesis is focused on problematic of purposes, content and methods of database thinking development. Experimental part proposes quantitative metrics for database thinking development.

KEYWORDS:
Education, Database, Database thinking, Structured Query Language

Obsah

1 Úvod

Na sklonku 20. století a na počátku 21. století jsou nové disciplíny ve vědě, technologiích a společnosti příčinou exponenciálního nárůstu objemu informací. Do centra zájmu se díky tomu dostává oblast lidského konání zaměřená na zprostředkování informace. V souvislosti s narůstající důležitosti informací ve společnosti tak mluvíme o příchodu informačního věku. Banks (Banks, 1993) konstatuje: „Stejně jako uvedení strojů v 19. století vedlo k mechanizaci společnosti, pokrok v elektronice a počítačích vyústil do role informací jako jednoho z nejdůležitějších druhů zboží 20. století. Spojení nových poznatků z elektroniky, počítačových technologií, ukládání dat, komunikace a zobrazovacích technik prostupuje společnost ve všech oblastech jejího konání. V posledních dekádách 20. století se pokrok v manipulaci s informací stal hlavním činitelem ohromného rozšíření technologií a vědy.

Nástup informační společnosti s sebou přináší i potřebu aplikovat prostředky informačních a komunikačních technologií (ICT) do každodenního života. Pokud chceme vyhovět nárokům s tím spojených, je nezbytně nutné, aby občan informační společnosti dokázal vstřebávat vzrůstající nároky na schopnost práce s ICT. Jejich objem se kontinuálně zvyšuje jako důsledek rozvoje existujících informačních produktů a služeb. Vyšší nároky informační společnosti jsou důsledkem nejenom složitější práce s informacemi, ale i růstu jejich objemu. Sin Cheung Konga (Kong & Li, 2009) uvádí tři základní příčiny výše uvedeného jevu. Prvotní příčina rostoucích nároků na informační gramotnost populace je exponenciální nárůst objemu informací. S tím je spojený pokrok v oblasti technologických postupů nezbytných ke zpracování stále většího objemu informací. Druhotná příčina spočívá v nárůstu obliby digitální kultury jako každodenního životního stylu. To vede k vývoji nových prostředků sloužících k implementaci pokročilých ICT technologiích do jednotlivých odvětví lidské činnosti. Třetí činitel má svou příčinu v ekonomické globalizaci. Rozvoj ICT dovedl občana informační společnosti k prostředkům, jež dovolily komunikovat a spolupracovat lidmi na planetě bez ohledu na geografické hranice.

V kontextu udržení kroku s tempem rozvoje ICT se do popředí zájmu informační výchovy dostává kromě samotné učení nových poznatků a dovedností, i podrobná analýza vnitřního uspořádání nově vznikajících nároků. Tu považuji za stěžejní úlohu současného rozvoje didaktiky informační výchovy. Proto je nezbytné oblast didaktiky informační výchovy flexibilně korigovat v reakci na změny a zaměřit se na formování kognitivních struktur žáka.

1.1 Informační společnost

V informační společnosti je běžnou součástí každodenního života schopnost a dovednost ovládat prostředky založené na databázových principech. Občan informační společnosti si často neuvědomuje, jaký prostředek ovládá, a databáze se staly transparentní součástí informační infrastruktury. Běžná, uživatelská práce s databází nevyžaduje žádné specializované nástroje a je ve většině případů dostupná intuitivními grafickými rozhraními. Grafické rozhraní pro manipulaci s databází může být v praxi realizováno například formulářem, rozbalovacím menu, seznamem položek atp. Charakteristické aplikace, jež na pozadí využívají databázi, jsou v každodenním životě občana například rozhraní pro vyhledávání titulů v katalogu knihovny, rezervaci letenek nebo hledání dopravních spojů.

Jak již bylo řečeno, frekvence používání ICT kontinuálně roste. Dokladem může být například rozšíření ICT do komunikace s úřady, kde již v některých případech může nahrazovat standardní doposud používané prostředky. Kromě toho lze očekávat další rozšíření hromadného nasazování ICT i v budoucnosti. To ve svém důsledku povede k dalšímu rozšíření oblasti informační gramotnosti.

Morální aspekt rozvoje informační gramotnosti mi slouží jako hlavní motivace pro snahu přispět k rozvoji etablující se didaktiky informační výchovy. Tím, že někteří lidé jsou informačně gramotní a někteří nejsou, vzniká jev nazvaný „digitální rozdělení“. To je podle Belangera (Belanger & Carter, 2009) „rozdíl mezi těmi, kteří k informacím přístup mají a kteří ne“. Tím vzniká rozdíl ve vzdělávání a vnímání lidí, který je rozděluje na počítačově gramotné a počítačově negramotné. Příčiny digitálního rozdělení jsou výslednicí nerovného přístupu k informačním a komunikačním prostředkům. Z toho vyplývá nedostatek znalostí a dovedností potřebných k jejich ovládnutí. Hlavní nástroj, metoda nebo také prostředek k minimalizaci dopadu digitální propasti napříč společností je rozvoj informační gramotnosti informační výchovou.

1.2 Informační výchova

Jak uvádí Průcha (Průcha et al., 2003), je výchova proces záměrného působení na osobnost člověka s cílem dosáhnout pozitivních změn v jejím vývoji. Různá pojetí výchovy byla ovlivněna sociokulturními podmínkami, odlišenými koncepcemi pojetí člověka, zdůrazněním jednotlivých stránek výchovného procesu. Někteří autoři chápou výchovu jako plně řízený proces ovlivnění nehotového člověka pedagogem nebo institucí, naplněný snahou podřídit jej normám společnosti, ale i normám instituce. Jiní zdůrazňují úlohu samotného vychovávaného jako subjektu formování a zvýrazňují podíl osobnosti na vlastním utváření (J. J. Rousseau, J. Dewey, E. Key, C. Freinet a další). Třetí proud vychází při vymezení výchovy z interakce mezi pedagogem a žákem (např. D. S. Peters aj.). Účinnost výchovy je závislá na míře interiorizace výchovných vlivů vychovávaným jedincem. K té dochází, je-li jedinec otevřen pedagogickému působení, odpovídá-li jeho zkušenosti, vytvoří-li se v jeho vědomí potřeby zdokonalovat se, stát se subjektem vlastního utváření. Neúčinnost čistě verbálního působení je nahrazována účinnější organizací zkušenosti vychovávaného. Z moderního hlediska je proto výchova především proces záměrného a cílevědomého řízení podmínek umožňujících optimální rozvoj každého jedince v souladu s individuálními dispozicemi a stimulujících jeho vlastní snahu stát se autentickou, vnitřně integrovanou a socializovanou osobností.

Cílem výchovy v ICT je záměrně a cílevědomě působit na člověka s cílem vytvořit vhodné osobnostní kvality, jež budou stabilním základem pro zvládnutí úlohy člověka na výzvy informační společnosti. Základním předpokladem k tomu je, aby člověk byl informačně gramotným. Informační výchova je hlavním prostředkem k budování informační gramotnosti. Jejím cílem je rozvíjet informačně technologické kompetence. V současném pojetí je informační výchova oporou rozvoje adaptability žáků, otevírá nové kontexty a klade důraz na schopnost učit se v dynamickém prostředí. (Rambousek & Štípek, 2007)

Etablující se didaktika informační výchovy zahrnuje schopnost a porozumění strukturaci poznatků, dekompozice a zpětné kompozice, vytváření, organizace a reorganizace datové struktury. Její klíčová úloha však spočívá v přípravě budoucích učitelů informačních a komunikačních technologií na jejich budoucí povolání. Stěžejní úkol informační výchovy je podle Bushe (Bush, 2009) rozvoj strategického, kritického, divergentního a kreativního myšlení. Mudrák (Mudrák, 2007) se domnívá, že současný výzkum v didaktice informační výchovy lze charakterizovat jako hledání cest efektivního začleňování ICT do výchovně-vzdělávacího procesu populace, který by nejenom podporoval hlubší pochopení principu

a zákonitostí potřebných pro život v informační společnosti, ale i napomáhal rozvíjet kognitivní dovednosti žáků. Hlavním cílem didaktiky informační výchovy je kultivace didaktického myšlení učitelů. Toho lze dosáhnout citlivým přístupem k vnímání základních pedagogických trendů v obecné didaktice, odmítáním jednostranné orientace na učivo v jeho encyklopedickém a vědeckém pojetí, překonáváním jednostranného vnímání racionalistického přístupu k pedagogickému procesu, odmítáním pasivního pojetí žáka a autoritativního klimatu 3školy a zdůrazňováním priority zaměření na vnitřní motivaci žáků (Skalková, 1999).

1.3 Databázové myšlení

Ve svém bádání jsem se zaměřil na dílčí oblast ICT. Tou je tematický celek databázové systémy. V něm se věnuji otázkou vymezení kompetencí pro práci s databází. Do nich jsem zahrnul nejenom čistě uživatelský přístup k používání databáze, ale i komplexnější pohled na pochopení vnitřního uspořádání databáze, porozumění databázovému jazyku a znalost principů systému řízení báze dat. Souhrnně pro databázové znalostí a dovedností zavádím pojem rozvoj databázového myšlení. Termín databázové myšlení je ve významu, používaném v práci širší, než je přímo jeho jazykový význam. Z obecného hlediska se jedná o specifický způsob manipulace s informacemi.

Pod rozvojem databázového myšlení v disertační práci souhrnně označuji soubor dovedností a znalostí vedoucích ke zvládnutí komunikace s datovou bází, schopnost navrhovat a implementovat vnitřní uspořádání datové báze a dovednost prezentovat údaje získané z datové báze. Specifickým elementem je v kompetenční skupině prezentace údajů získaných z databáze nutné brát v úvahu i dovednost pracovat dotazovacím nástrojem databáze. Ve vztahu k informační gramotnosti řadím databázové myšlení do širšího kontextu hledání údajů, schopnosti ověřit autentičnost informace a evaluace stupně informační relevance v komplexním kontextu technologického prostředí, viz např. Bush (Bush, 2009).

1.4 Cíl práce

Disertační práce obhajuje tezi integraci databázového způsobu uvažování do základů informačně technologických kompetencí. Pojem databázové myšlení je v disertační práci charakterizován jako komplexní soubor činností, znalostí a principů, které dávají možnost zvládnout práci s databází, návrh struktury databáze a prezentaci údajů z databáze. Oproti tomu základy práce s databázovými systémy jsou učiteli v České republice, jak uvádí ve své publikaci Rambousek a kolektiv (Rambousek & Štípek, 2007), vnímány pouze s marginálním zájmem. Podle výsledků výzkumu by databáze 97 % učitelů obětovalo ve prospěch jiné kompetence informačních a komunikačních technologií. Při analýze zbytných celků informačních a komunikačních technologií jsou základy práce s databázovými systémy nebo tvorba databází zařazeny do kandidátské skupiny nejvýše. Analogicky je práce s databázovými systémy na druhém místě nejméně oblíbených tematických celků pro suplování. Výsledky výzkumu zveřejněné Neumajerem (Neumajer, 2007) výše uvedená konstatování potvrzují. Ve své práci se zabýval názorem učitelů na znalosti a dovednosti nezbytné pro vstup do pokročilé úrovně školení informačních a komunikačních technologií. Učitelé považují téma databáze za nezbytný předpoklad pro vstup do pokročilého modulu pouze v 0,1 % případů. Stejný jev lze pozorovat i u českých učitelů při pohledu na databáze jako na potenciální oblast dalšího profesního rozvoje učitelů. Pouze 0,55 % z celkového počtu respondentů se domnívá, že by databáze měly být rozvíjeny v přímé návaznosti na úvodní modul pokročilé úrovně.

I přes současný stav věcí v České republice považuji schopnost návrhu, implementace a práce s databází za základní a neoddělitelnou oblast ICT. Z toho pramení motiv celé práce, jež směřuje k implementaci didaktiky databázových technologií do jádra didaktiky informačních a komunikačních technologií. Základním prostředkem k analýze cíle databázového myšlení je rozbor existujících standardů.

V oblasti analýzy obsahu výuky databázového myšlení využívám primárně poznatky o psychologických aspektech výuky programování. Tím dostávám podklad pro konstrukci kognitivního modelu rozvoje databázového myšlení založeného na principu stabilních prvků znalosti. To poskytuje dobrý základ pro systematizaci obsahu databázového myšlení a umožňuje charakterizovat hierarchii znalostních prvků. Po jejím stanovení je možné rozborem vnitřní struktury určit optimální pořadí pro jejich osvojení.

Teoretická část práce se skládá ze tří hlavních celků. První část je věnována cílům výuky

databázového myšlení, druhá část obsahu rozvoje databázového myšlení a třetí část metodám rozvoje databázového myšlení. Konkrétní hierarchie hlavních cílů a z nich vyplývajících úkolů je následující:

1) Stanovit cíl rozvoje databázového myšlení.
    a. Definovat obecné cíle rozvoje databázového myšlení.
    b. Sestavit přehled a systematizovat cíle rozvoje databázového myšlení.
2) Strukturovat obsah rozvoje databázového myšlení.
    a. Stanovit rámec analýzy obsahu rozvoje databázového myšlení.
    b. Uvést konkrétní obsah rozvoje databázového myšlení.
    c. Systematizovat obsah rozvoje databázového myšlení.
3) Uvést metody výuky databázového myšlení.
    a. Rozpracovat kognitivní model rozvoje databázového myšlení.
    b. Identifikovat aplikační domény kognitivního modelu rozvoje databázového myšlení studentů.
    c. Podat přehled teoretických základů didaktických prostředků databázového myšlení.
    d. Vytvořit koncept kvantitativních charakteristik rozvoje databázového myšlení.
4) Analyzovat charakteristiky rozvoje databázového myšlení studentů.
    a. Kvantifikovat rozdíl mezi použitými výukovými metodami.
    b. Porovnat rozdíly ve výkonnosti studentů vyčíslením.
    c. Identifikovat problematické celky sledováním průběhu chybovosti.
    d. Určit existenci vztahu mezi četností a chybovostí studentů.
    e. Kvantifikovat vztah mezi četností a chybovostí studentů.
    f. Detekovat vstupní faktory s vlivem na schopnost učení studenta.

Uvedená struktura slouží jako osnova pro orientaci v práci. Její struktura je založena na didaktickém zpracování tématu a praktická část je zaměřena na analýzu rozvoje databázového myšlení u studentů učitelství ICT.

2 Cíl rozvoje databázového myšlení

Při hledání směru výuky databázového myšlení bylo přistoupeno syntéze existujících výukových cílů dle mezinárodních standardů. Na jejich základě byl jako hlavní cíl utváření databázového myšlení určen rozvoj znalostí dotazovacího jazyka. Stěžejním prvkem v něm hraje roli formování odpovídající znalostní struktury žáka.

2.1 Cíl výuky databázového myšlení

Skalková (Skalková, 1999) definuje cíl vyučování jako zamýšlený a očekávaný výsledek, k němuž učitel v součinnosti se žáky směřuje. Tento výsledek je vyjádřen ve změnách, jichž se prostřednictvím vyučování dosahuje ve vědomostech, dovednostech. Kalhouse (Kalhous & Obst, 2002) definuje kontrolovatelnost výukových cílů. Ta je určena tím, zdali se nám prostřednictvím vyučování podařilo navodit učení žáka, tedy zda došlo k zamýšleným změnám v jeho kompetencích (znalostech, dovednostech, postojích). To můžeme posoudit jen na základě pozorovatelné činnosti žáků. Proto má vymezení cílů vyjadřovat, jaké činnosti, jakého výkonu je žák schopen v určité etapě svého učení dosáhnout.
Za stěžejní cíle rozvoje databázového myšlení považuji kromě praktické roviny osvojení kompetencí pro práci s databází i formování odpovídající kognitivní struktury žáka. Pro lepší orientaci ve hierarchii cílů jsem je rozdělil do kompetenčních skupin. Konkrétně se jedná o oblasti:

1) znalosti sémantiky a syntaxe dotazovacího jazyka,
2) vnitřního uspořádání datové báze,
3) dovedností pracovat s datovými formuláři,
4) modifikovat databázovou strukturu
5) produkovat výstupní sestavy.

Dílčí cíle rozvoje databázového myšlení poté jsou:

1) Schopnost rozpoznat práci s databází v rámci každodenní činnosti.
2) Schopnost chápat databázi jako nástroj pro práci s informacemi.
3) Znát databázi a rozumět vnitřnímu uspořádání databáze.
4) Zvládnout práci s databází.

Jednou z autorit, která se věnuje ICT kompetencím je Organizace pro kvalifikaci a kurikulum (Qualification and Curriculum Authority, QCA). Ta vydala dokument (QCA, 2003) popisující standard platný pro vzdělávací instituce ve Velké Británii. Dokument má podobu metodické příručky a obsahuje stěžejní principy rozvoje ICT kompetencí. Jejím hlavním úkolem je pomoci učiteli při sestavování obsahu výuky.

Standard je součástí národního kurikula Velké Británie. Je formulován cílovými kompetencemi, které by žáci měli učením dosáhnout. Ve vztahu k tématu disertační práce jsou ve standardu relevantní části věnované databázím. Podrobněji jsou jednotlivé kompetence uvedeny v příloze 10.2 Databázové kompetence dle QCA.

Databázím je v QCA věnován nácvik následujících praktických dovedností:

Úroveň	**Popis**
1) Úroveň	Použití slovní zásoby
2) Úroveň	Informace kolem nás Charakteristika a třídění Hledání informací Otázky a odpovědi
3) Úroveň	Úvod do databází
4) Úroveň	Větvené databáze
5) Úroveň	Analýza údajů
6) Úroveň	Vyhledávání na Internetu

Tabulka 1 Úroveň praktických dovedností v oblasti databází dle QCA

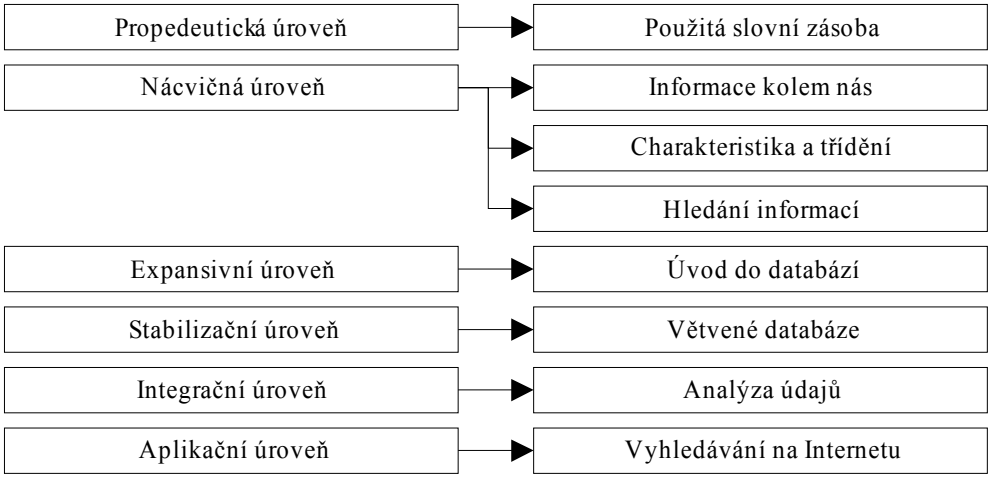
Na základě bližší analýzy struktury britského kurikula jsem stanovil tří základní úrovně rozvoje databázového myšlení. Při rozvoji databázového myšlení je nezbytné zachovat pořadí a posloupnost nácviku jednotlivých dovedností. Konkrétně jsem identifikoval následující úrovně:

1) Propedeutická úroveň – hlavním cílem je položit základ strukturovaného myšlení a rozvoj organizace dat.
2) Nácvičná úroveň – klíčovým tématem úrovně je rozvoj schopností pro korektní formulaci dotazů a kognitivních předpokladů pro určení dotazovacího jazyka. Dále schopnost představy vnitřního uspořádání datové báze a zvládnutí dalších pracovních úloh směřujících k práci s databází.
3) Expansivní úroveň – jedná o vstupní úroveň, která postihuje základy databázových

principů

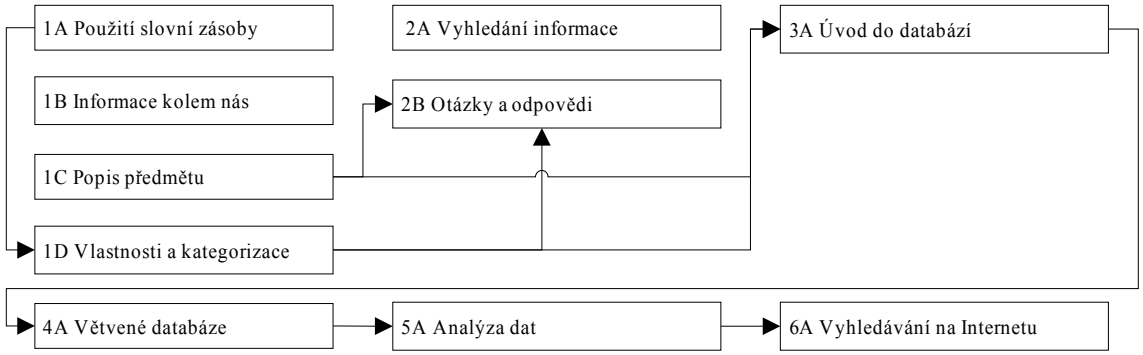
4) Stabilizační úroveň – klíčová úroveň pro rozvoj kognitivní základny pro utváření vnitřního uspořádání databáze.
5) Integrační úroveň – v integrační úrovni je cílem nabyté znalosti integrovat a propojit s cílem připravit odpovídající znalosti pro aplikační úroveň.
6) Aplikační úroveň – stěžejním úkolem úrovně je aplikovat nabyté znalosti do reálného prostředí práce s databází.

Syntézou obsahu jednotlivých praktických dovedností vzniká struktura pro jejich rozdělení do skupin. Na diagramu jsou znázorněny úrovně rozvoje základů databázového myšlení:



Obrázek 1 Úrovně rozvoje databázového myšlení

Vnitřní uspořádání a návaznost jednotlivých dovedností je přesně dáno a lze jej vyjádřit následujícím grafem:



Obrázek 2 Návaznost etap rozvoje databázového myšlení dle QCA

Rozvoj cílů databázového myšlení lze vyjádřit také výčtem cílových kompetencí, jichž musí žák dosáhnout. Do databázového myšlení zahrnuji porozumění konceptů databází, schopnost formulovat dotazy v dotazovacím jazyce a znalost provozu databázových systémů. Do oblasti dovedností databázového myšlení zahrnuji i zadávání údajů do databáze, schopnost navrhnout strukturu databáze a znalost pracovního postupu pro tvorbu výstupních sestav z údajů uložených v databázi. Ve znalostní oblasti do databázového myšlení řadím i porozumění principům činnosti databázových systémů, schopnost vyjádřit dotaz v dotazovacím jazyce, přehled o konkrétním databázovém paradigmatu (hierarchická, relační, objektová databáze) a schopnost porozumět datovému modelu. Z uživatelsko-aplikačního hlediska do rozvoje databázového myšlení zahrnuji znalost formulářů, schopnost práce s nimi a schopnost vyčíst ze sestavy potřebné informace. Za dílčí cíl rozvoje databázového myšlení považuji schopnost rozpoznat práci s databází v rámci každodenní činnosti. Znamená to schopnost uvědomit si, že vyhledávání v knihovním systému je ve své podstatě zadávání omezujících kritérií dotazu do formuláře, z nějž je vykonstruován dotaz položený databázovému systému atp.

Kromě vertikálního členění kompetencí dle jejich náročnosti jsem identifikoval tři kompetenční skupiny. Jejich členění je založeno na rozboru zákonitostí pracovních postupů používání a návrhu databáze. Dílčí kompetenční oblasti jsou seskupeny podle cíle, kterým lze jejich osvojení dosáhnout: sběr dat, návrh struktury datové báze a prezentace informací tvorbou sestavy. Základem pracovních postupů ve výše uvedených skupinách je princip práce s informační strukturou, a to jednak: získávání informací, definice vnitřního uspořádání informací a způsob předávání informací. Obecně se tedy jedná o příjem, zpracování a výstup informací. Výše uvedené členění bere ohled na kompetenční skupiny založené na principech rozkladu jednotlivých částí pracovních postupů spadajících do databázového myšlení.

Pořadí a návaznost jednotlivých fází není v rámci práce jasně definováno. Hlavní důvod je právě neprocedurální přístup k práci s databází, ve kterém mohou být oblasti pracovních postupů vykonávány bez ohledu na pořadí. Pokud bychom uvažovali volné stanovení pořadí výše uvedených činností, bylo by asi následující: tvorba struktury databáze, vkládání údajů do databáze a prezentace údajů z databáze. To však platí pouze pro první iteraci a v navazujících iteračních cyklech již pořadí nemusí být dodrženo.

Jiný pohled na rozdělení cílů databázového myšlení lze nalézt aplikací kategorizace dle Boyce (Boyce & Chamberlin, 1974):

1) Deklarativní znalost – to jest znalost principu fungování, porozumění důvodů k chování, znalost názvu principů a znalost pozice v komplexním systému. Do

kategorie deklarativní znalosti patří také informaci o konceptech, elementech a vnitřních vazbách znalosti s ohledem na vazby na konkrétní subjekty.

2) Procedurální znalosti – to je detailní znalost úseků pracovních postupů nebo aktivit, které jsou zapotřebí k dokončení úlohy anebo práce. Jednotlivé kroky procesů jsou součástí ucelených celků procesů a sledováním jejich posloupnosti lze dovolit provedení následující úlohy bez rozhodovacího procesu, zdali povolit provedení akce.
3) Strukturální znalost – zabývá se reprezentací vnitřního uspořádání poznatků v deklarativní oblasti (kategorie pojmů), sémantickými sítěmi, schématy zaměřenými na řešení úlohy, scénáře a reprezentací obsahů v procedurální paměti (produkční systém).

Výše uvedené členění struktury znalostí lze aplikovat i na elementy znalostí databázového myšlení. Na základě kritérií pro každou kategorii lze kompetence rozřadit do jednotlivých skupin.

1) Deklarativní oblast databázového myšlení zahrnuje porozumění principům databázových systémů a schopnost formulovat elementy dotazovacího jazyka.
2) Procedurální oblast databázového myšlení obsahuje úseky pracovních postupů nebo celé pracovní postupy související s používáním aplikačního rozhraní databáze, pracovní postupy návrhu databázové struktury a kognitivní procesy probíhající při formulaci dotazu v dotazovacím jazyce.
3) Strukturální oblast databázového myšlení je charakteristicky determinována znalostí datového a konceptuálního modelu, obecnou znalostí hierarchie pojmů databázového myšlení a znalosti syntaxe dotazovacího jazyka.

Na obecné rovině práce s informacemi patří databázové myšlení do oblasti organizace informací. Jedním ze zdrojů pojednávajících o obecných konceptech informačních věd je například NCVER (NCVER, 2003). Ten do širšího kontextu kategorie organizace informací řadí sběr a organizaci informací, řešení problémů, plánování a organizování, učení se učit, inovativní myšlení, kreativní systémové myšlení.

Jako další se organizací informací zabývá Mayer (Mayer, 1992). Ten ji považuje za klíčovou kompetenci a v širším kontextu spatřuje ještě další kategorie: schopnost získat, analyzovat a organizovat informace, schopnost předávat myšlenky a informace, dovednost plánovat a organizovat pracovní aktivity, dovednost spolupracovat s ostatními a práce v týmu,

schopnost používat systémové postupy a metody, schopnost zvládnout logické úvahy a strategie řešení problémů, dovednost využívat technická zařízení.

V rámci pokusu o širší pohled na problematiku organizace práce jako východiska pro stanovení okolních vazeb databázového myšlení jsem se zabýval i pohledem, který na organizaci informací zaujímá psychologie. Vycházím z definice v Encyklopedii aplikované psychologie – „Encyclopedia of Applied Psychology“ (Rafoth, 2004). Ten řadí organizaci informací do oblasti procesu kognitivního zpracování. Proces je v tomto pojetí chápán jako operace, která buď vytváří, nebo transformuje myšlenkovou reprezentaci. Může se tedy jednat o překlad vnějšího vstupu do konkrétní reprezentace, modifikaci reprezentace jako takové nebo vytváření výstupu. Proces je tedy v tomto pojetí funkce, která spojuje vstup a výstup. V rámci modelů kognitivního zpracování existuje spojení jednoduchých jednotek do vrstev, čímž je umožněna organizace informací v rámci fází zpracování.

Definicí organizace informací se zabývá i Organizace pro ekonomickou a hospodářskou spolupráci OECD (OECD, 2005). Ta řadí organizaci informací jakou součást skupiny kompetencí související s interaktivním používáním nástrojů. Jejich ovládnutí považuje za důležitou kompetenci, jež může být základem pro osvojení schopnosti interaktivního používání technologií. V rámci výše uvedené definice se jedná o upřesnění požadavku na organizaci znalostí a informací.

Domnívám se, že při zkoumání metod rozvoje databázového myšlení je nezbytné zařadit jej do širšího kontextu obecné soustavy klíčových kompetencí. Názory jednotlivých autorů na přesnou definici klíčových kompetencí se liší (Belz et al., 2001), (Rada, 2006), (VÚP, 2007a). Pro účely zasazení databázového myšlení do kontextu klíčových kompetencí jsem vybral kategorii související se získáváním, analýzou a organizací informací. Domnívám se, že výše uvedené kompetence mohou být základem pro rozvoj schopností informace vyhledat, prověřit a kategorizovat. Klíčová kompetence rozvoje prezentace údajů a jejich přetavení do užitečné formy je přínosná jak z hlediska kompetenčního rozvoje, tak i z hlediska možnosti zhodnocení získaných informací. Jak znalost informačního zdroje, tak i metody pro dobývání znalostí z informačních zdrojů jsou primárním kompetenčním předpokladem pro zvládnutí databázového myšlení. Rozvoj výše uvedených předpokladů může být realizován například i neprofesionálním používáním knihovnicko-informačních služeb.

Jak jsem již předeslal, databázové myšlení lze v obecné rovině uchopit jako specifický prostředek či metodu pro organizaci informací. Obdobné pojetí oblasti využití databázového myšlení deklaroval i Evropský parlament v dokumentu „Doporučení Evropského parlamentu a Rady ze dne 18. prosince 2006 o klíčových schopnostech pro celoživotní učení“ (Rada,

2006). Dle mého názoru se tak rozvoj databázového myšlení řadí do oblasti získávání, ukládání a vytváření informací. Tematicky spadá do celku rozvoje schopností práce s digitálními technologiemi. V dokumentu se konkrétně uvádí: „… kritické používání technologií informační společnosti při práci, ve volném čase a v komunikaci. Předpokladem je základní znalost informačních a komunikačních technologií, tj. používání počítačů k získávání, hodnocení, ukládání, vytváření a výměně informací a ke komunikaci a spolupráci v rámci sítí prostřednictvím internetu. Nejdůležitější znalosti, dovednosti a postoje související s touto schopností. Pro schopnost práce s digitálními technologiemi jsou nezbytné důkladné pochopení povahy, úlohy technologií informační společnosti a jejich možností v každodenních situacích a důkladné znalosti z těchto oblastí v osobním a společenském životě i v práci“. Ve stejném dokumentu Rady (Rada, 2006) je možné nalézt i specifikaci dovedností, zahrnujících „schopnost vyhledávat, shromaždovat a zpracovávat informace a používat je kritickým a systematickým způsobem, hodnotit jejich důležitost a rozlišovat mezi reálnými a virtuálními informacemi a zároveň chápat vztahy. Jedinci by měli umět používat nástroje k vytváření, prezentaci a pochopení komplexních informací a měli by být schopni používat služby počítačové sítě internet pro získání, vyhledání a používání. Rovněž by měli umět používat technologie informační společnosti k podpoře kritického myšlení, tvořivosti a inovací. Pro využívání technologií informační společnosti je nezbytný kritický a reflexivní postoj k dostupným informacím a odpovědné používání interaktivních médií. Schopnost je rovněž rozvíjena zájmem o zapojení se do kolektivů a sítí pro kulturní, sociální nebo profesní účely.“ Konkrétně je potom rozvoj databázového myšlení zmíněn ve stejném dokumentu a je v něm definován jako součást „základních počítačových aplikací, např.: textové editory, tabulkové procesory, databáze, systémy ukládání a správy informací, pochopení možností a potenciálních rizik, jež internet a komunikace prostřednictvím elektronických médií přinášejí pro práci, volný čas, sdílení informací a spolupráci, učení a výzkum v rámci sítí“. Z toho vyplývá, že databáze je i Evropskou radou považována za základní počítačovou aplikaci. Z dokumentu jasně vyplývá, že její pochopení je nutné pro úspěšné uplatnění v informační společnosti.

Na základě výše uvedených skutečností se domnívám, že v obecné rovině je nutné databázové myšlení směřovat nejenom do oblasti rozvoje konkrétních dovedností, znalostí a postojů, ale sledovat i nadřazené a okolní kompetence. Hlavní klíčové kompetence, které považuji za nezbytné rozvíjet i při rozvoji databázového myšlení, spadají do oblasti práce s informacemi. Jedná se o kompetence související s vyhledáváním, shromažďováním, zpracováváním a hodnocením informací. Kromě toho do skupiny příbuzných kompetencí patří i schopnost

informace organizovat, reorganizovat, uspořádávat a třídit.

Ke konkrétnímu naplňování vzdělávacích cílů jsou v České republice určeny Rámcové vzdělávací programy. Konkrétně se jedná o Rámcový vzdělávací program pro předškolní vzdělávání (RVP PV) (VÚP, 2004), Rámcový vzdělávací program pro základní vzdělávání (RVP ZV) (VÚP, 2007c) a Rámcový vzdělávací program pro gymnázia (RVP G) (VÚP, 2007b).

Cíle základního vzdělávání jsou z pohledu rámcových vzdělávacích programů položeny do roviny postupného rozvíjení klíčových kompetencí tak, aby poskytly spolehlivý základ všeobecného vzdělání orientovaného zejména na situace blízké životu, uplatňování svých práv a naplňování svých povinností. Jejich podrobnější analýza však přesahuje rámec této disertační práce. V následujících odstavcích uvedu pouze styčné plochy mezi databázovým myšlením a podle mého názoru jejich nejužší vazby na téma rozvoje databázového myšlení. Z tohoto pohledu může být databázové myšlení charakterizován jako realizace tvořivého myšlení, logického uvažování a řešení problémů. Lze z nich vycházet při stanovování dalších relevantních klíčových kompetencí, jež vedou žáka k:

1) Porozumění toku informací, počínaje jejich vznikem, uložením na médium, přenosem, zpracováním, vyhledáváním a praktickým využitím.
2) Schopnosti formulovat svůj požadavek a využívat při interakci s počítačem algoritmické myšlení.
3) Žák umí pracovat s informačními zdroji.
4) Dokáže používat rejstřík, obsah a v případě elektronického zdroje i vyhledávání.
5) Dokáže posoudit relevanci informace z hlediska její použitelnosti.
6) Dokáže dekomponovat problém na menší problémové celky.
7) Dokáže integrovat výsledky dílčích řešení.
8) Dokáže formulovat výsledek v podobě řešení problému.

Výše uvedené citace měly za cíl uvést některé autority nebo prameny za účelem zasazení databázového myšlení do kontextu informační výchovy a vzdělávací soustavy obecně.

Definice cílů rozvoje databázového myšlení v rámci RVP ZV není explicitně uvedena. Oproti tomu výše zmiňovaný standard QCA deklaruje konkrétní činnosti, které mohou vést k rozvoji databázového myšlení již na úrovni prvního stupně základního vzdělávání. Na úrovni základního vzdělávání by bylo vhodné zapracovat rozvoj databázového myšlení propedeutické v podobném rámci, jako jej deklaruje QCA. Na středoškolské úrovni odpovídá

rozsahu potřebných znalostí ECDL. V přípravě budoucích učitelů ICT by bylo vhodné zaměřit se na didaktické prostředky pro rozvoj databázového myšlení. Cíl rozvoje databázového myšlení by však neměl být definován na kompetenční úrovni, ale jako určitý stupeň rozvoje kognitivních dovedností v oblasti práce s databází s důrazem na znalost dotazovacího jazyka.

2.2 Kognitivní model znalosti dotazovacího jazyka

Cen (Cen et al., 2006) popisuje kognitivní model znalostí dotazovacího jazyka v oblasti rozvoje inteligentních vyučovacích systémů jako množinu produkčních pravidel nebo dovedností sloužících k predikci způsobu, jakým budou studenti řešit problém. Běžně takový model vzniká na základě spolupráce vědců a oborových odborníků. Cen navrhnul částečně automatizovanou metodu pro konstrukci přesnějšího kognitivního modelu, ve které využil metodu faktorové analýzy a dal jí pracovní název „faktorová analýza učení“. Metoda kombinuje statistický model, lidský úsudek a kombinatoriku. Může sloužit k evaluaci existujících kognitivních modelů nebo generování a evaluaci alternativních modelů. Vychází z přesvědčení, že kognitivní model je množina pravidel a dovedností zakomponovaná v inteligentním vyučovacím systému proto, aby modelovala způsob řešení problémů studenty. Kvalitní kognitivní model zachycuje detailně znalostní komponenty v kurikulu a poskytuje zpětnou vazbu a nápovědu, vybírá problémy k řešení v závislosti na potřebě jednotlivých studentů, pomáhá studentům dosáhnout lepších učebních výsledků. Při konstrukci kognitivního modelu si Cen položil následující otázky:

1) Jak popsat žákovy myšlenkové pochody stávající terminologií kognitivního modelu?
2) Jakým způsobem lze efektivně evaluovat validitu evoluce kognitivního modelu?
3) Jak použít získané znalosti ve výuce?

Domnívám se, že otázky formulované Cenem jsou vyjádřením jádra tematické náplně výzkumu kognitivních modelů rozvoje databázového myšlení.

Kognitivním modelem programování se zabýval už Mayer (Mayer, 1975). Zveřejnil tři modely procesu znalostí databázového myšlení. Jednotlivé modely se od sebe liší počtem proměnných vstupujících do procesu osvojování databázového myšlení. V jednostupňovém modelu učení je tak jedinou proměnnou množství vědomostí přijaté studentem. Míra pozornosti studenta tvoří hlavní faktor determinující výsledek jeho učení. Vnější instrukční

proměnná je kvantitativním vyjádřením množství informací zapamatované studentem. Jedná se o závislou proměnnou, která se často objevuje ve výzkumech verbálního projevu studenta. Ty kladou důraz na délku a frekvenci prezentace, opakování nebo pokyny pro dodržování kázně. Jednostupňový model předpokládá determinaci efektivity výuky ovlivněnou kvantitativním množstvím procvičovaného materiálu. Kromě toho vychází z předpokladu, že mezi pozorností studenta ve výuce a množstvím procvičované látky existuje přímá úměrnost. Jednostupňový model předpokládá zjištění vyšší kvantity přijímaných informací na základě rozboru výsledků kontrolního testu. Ten měří rychlost vybavování a rozpoznávání. Dvoustupňový model zahrnuje do procesu osvojování databázového myšlení kromě proměnné kvantitativního množství vědomostí přijatého studentem druhou proměnnou. Tou je množství existujících znalostí uložených v paměti studenta. Mayer považuje přijetí nové informace z procesního hlediska jako rozšíření stávající hierarchické struktury sekvence znalostí ve studentově paměti. Ve dvoustupňovém modelu se nabízí možnost výsledek učení měřit kvalitativně. Toho lze dosáhnout sledováním vnitřních proměnných, jež mají vliv na zpracování informace. Tím může být pozornost studenta nebo míra, se kterou student konfrontuje nově nabyté znalosti s již zapamatovanou bází poznatků. Model bere v potaz i externí proměnnou. V tom případě je studentova znalostní banka charakterizována jako množina zkušeností, jež si student v minulosti osvojil.

Model tedy vyjadřuje přesvědčení, že čím větší objem znalostí a zkušeností má student, tím větší je celkový objem toho, co je schopen se nově naučit. Z toho logicky plyne determinanta ovlivňující studentův výkon. Třístupňový model přidává do procesního modelu osvojování databázového myšlení kromě proměnné kvantitativního množství vědomostí přijatých studentem a množství znalostí, jež si student pamatuje, i hledisko charakteru procesu aktivace znalostí studentem během učení. Tím je myšlena oblast studentovy znalosti, jež může tvořit asimilační množinu připravenou pro integraci nových poznatků. Jejich úlohu považuje třístupňový model osvojování databázového myšlení za důležitou, neboť je determinantou, zdali je studentovo učení smysluplné, nebo mechanické. Dvoustupňový model umožnil pohlížet na výsledek učení nejenom z pohledu kvantity přijatého materiálu, ale i asimilační znalostní množinou, jež umožňuje asimilaci nových poznatků. Třístupňový model rozšiřuje koncept učení vložením kvantitativní hlediska, jako je objem znalostí, a zároveň přidává kvalitativní aspekt vnitřního uspořádání znalostí studenta. Hlavní proměnné ve třístupňovém modelu jsou vnímání materiálu, existence znalostních předpokladů a aktivace asimilační množiny. Externí instrukční proměnná vstupující do třístupňového modelu je podmíněna nejenom pořadím instrukcí, ale i důrazem na jednotlivé instrukce.

Mayer ve své další práci kognitivní modely databázového myšlení zcela neopustil. V roce 1981 (Mayer, 1981) se zabýval okolím procesu praktického využití databázového myšlení. To zkoumal v kontextu interakce s okolním světem a v kontextu práce s pamětí. Analyzoval vliv externích faktorů na rozvoj databázového myšlení v souvislosti s množstvím znalostí zapamatovaných studentem. Domníval se, že v krátkodobé paměti dojde ke spojení přijatého poznatku přijímaného se znalostí vyvolanou z dlouhodobé paměti. Tím může být student schopen formulovat zpětnou vazbu. Tu potom předat svému okolí jako odpověď. Mayer svoji teorii postavil na domněnce, že znalosti jsou v dlouhodobé paměti uloženy, aniž by byla omezena doba jejich platnosti. V kontextu rozvoje databázového myšlení je dlouhodobá paměť využívána hlavně při vybavování struktury pro uložení informací syntaktické a sémantické stránky dotazovacího jazyka.

V mysli studenta existují současně ale i oblasti, do kterých jsou ukládány vizuální, audiální, kinestetické a permanentní informace. Každá z těchto oblastí má odpovídající asociační kortex. Ten informaci drží, dokud není zapomenuta, poslána do pracovní paměti nebo uložena zpět do dlouhodobé paměti. Součinnost mezi krátkodobou a dlouhodobou pamětí je využita nejenom při překladu zadání problému v přirozeném jazyce do dotazovacího jazyka, ale i při dekódování výstupních symbolů. Nové informace vstupují do mozku pomocí smyslů. Ti, kteří preferují vizuální informace, mohou mít více sítí ve vizuálním kortexu než ti, kteří dávají přednost audiální informaci. Informace je získána z oblasti mysli, kde je uložena a poté přivedena do odpovídajícího asociačního kortexu. Informace vyvolaná z oblasti mysli, ve které je uložena, je poslána na odpovídající asociační kortex. Vědomí informaci může odmítnout nebo ji poslat do pracovní paměti, kde je seřazena a spojena s ostatními informacemi. Výše popsaný princip podle Mayerova názoru představuje model procesu osvojování nových poznatků, které umožňují rozvoj databázového myšlení.

Rozdělení znalostí dotazovacího jazyka jako klíčového prvku rozvoje databázového myšlení potvrzuje i Shneiderman (Shneiderman et al., 1977). Podle něj hraje sémantická znalost důležitou úlohu ve stylu řešení problémů, zatímco syntaktická znalost reflektuje hluboké porozumění problémových situací a schopnost modulární dekompozice. Sémantická znalost oproti tomu umožňuje rozlišení validity daného příkazu, vyhledání chyb v programu nebo vytvoření myšlenkového modelu procesu jeho vykonávání. Syntakticko-sémantický model chování může být aplikován na chování učitele při výuce dotazovacího jazyka a v oblasti didaktiky informačních a komunikačních technologií je důležitým nástrojem při formulaci pedagogických strategií výuky programování. Shneiderman dále popsal model spirálového přístupu k výuce programování. Formuloval výukovou metodu založenou na teorii pomalého

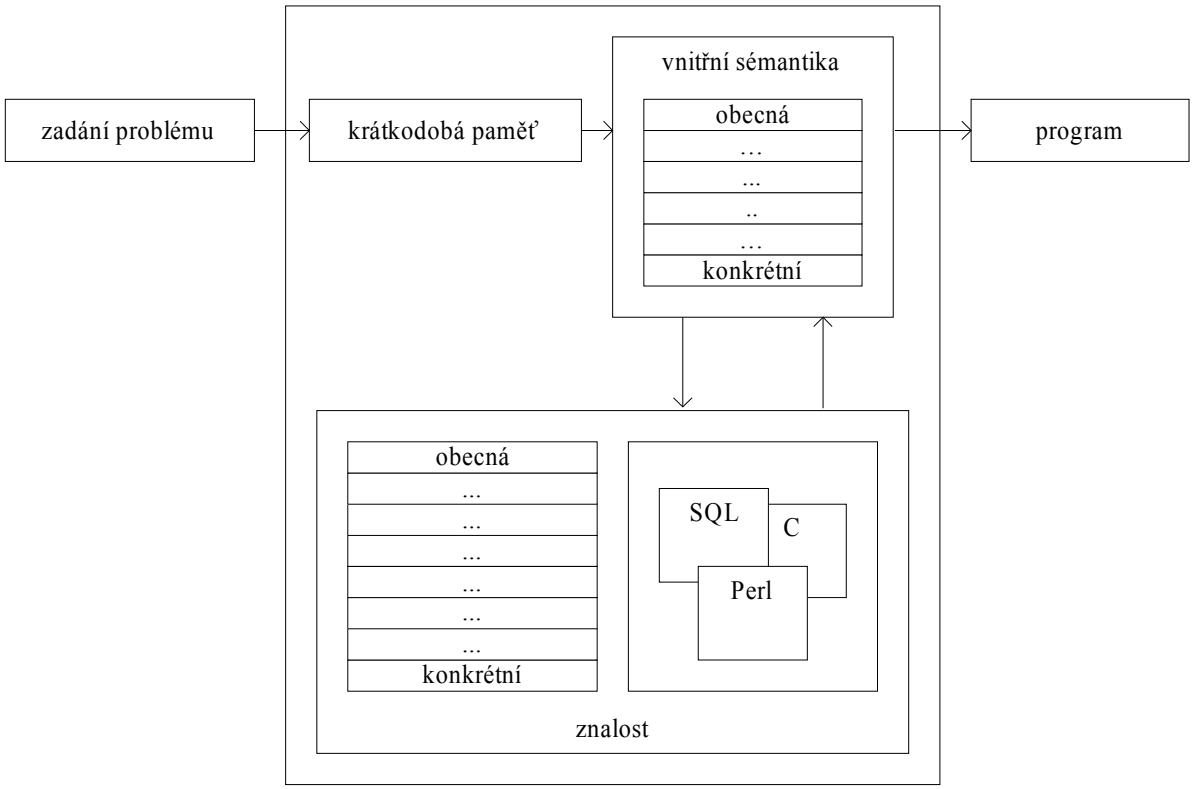
zvyšování zátěže studenta v koordinaci s postupným osvojováním syntaktické a sémantické znalosti, která klade důraz na charakter jeho znalostí. Syntaktickou znalost si je možné osvojit procvičováním a opakováním. Sémantická znalost je odolná vůči zapomínání a zapamatování nového sémantického poznatku a probíhá pouze smysluplným učením. Podle jeho názoru je ve výuce sémantické znalosti nezbytné postupovat při výkladu po malých částech, jež přesně pasují do struktury již osvojených znalostí.

Shneiderman se domníval, že nejefektivnější přístup k výuce dotazovacího jazyka je založený na rozvoji sémantických a syntaktických elementů. Zavrhnul přístup jednostranného syntaktického zaměření, jaký je znám z příruček programovacích jazyků. Nepovažoval za efektivní ani prezentaci složitých programů hned první den výuky. Ty se vyznačují předložením složitého počítačového programu studentům hned na začátku. Na jeho základě si studenti mají sami intuitivně odvodit správnou syntaxi jazyka, kontrolní struktury programu a programový design. Podle Shneidermana však výše uvedená vyučovací metoda není efektivní, protože nedbá na cílený rozvoj sémantické znalosti a nedodržuje pravidlo postupného zvyšování zátěže.

Shodný pohled na syntaktickou znalost zaujal i Wilson (Wilson & Keil, 2001). Považuje ji za vyjadřovací prostředek pro zachycení jazykové struktury vět. Jednou z jejích vlastností je teoretický podklad pro definice rozdělení vět do frází. Fráze jsou složeny z částí pojmenovaných lexikální kategorie (sloveso, podstatné jméno apod.). Pro rozvoj databázového myšlení je ovšem důležitý poznatek, který považuje za optimální podobu rozvoje syntaktické znalosti memorování. Protože je však syntaktická znalost předmětem zapomínání, musí být často obnovována. Sémantická znalost programovacího jazyka je oproti tomu znalostí programové struktury a algoritmů. Má hierarchickou strukturu a je rozvrstvena podle úrovně abstrakce. Nejnižší úroveň sémantické znalosti procedurálního jazyka je například porovnání dvou hodnot nebo operace přiřazování. Na střední úrovni je to sémantický princip funkce vyhledávání prvku v datovém poli s nejvyšší hodnotou nebo změna každého prvku datového pole na stejnou hodnotu. Vyšší sémantická úroveň používá nižší vrstvy. To lze ukázat na příkladě algoritmu pro řazení prvků, který používá sémanticky nižší funkce, jako je porovnání a přiřazování hodnot. Vyšší úrovně sémantické znalosti mají podobu procesu kompilace nebo transakčního zpracování. Pro sémantickou znalost je specifická odolnost vůči zapomínání a nezávislost na konkrétním programovacím jazyce. Syntaktická struktura znalosti programovacího jazyka je složena z podmiňování, kontroly toku a datových typů programovacího jazyka.

Shneiderman (Shneiderman et al., 1977) zkonstruoval model kompozice programu

rozložením a analýzou struktury programovacího jazyka na syntaktickou a sémantickou složku. Model znázorňuje proces kompozice od uchopení problému přes model kognitivních procesů až po sestavený program. V první fázi modelu je vstupní zadání problému uloženo do krátkodobé paměti. Na zadání v krátkodobé paměti je aplikována znalost z dlouhodobé paměti, a tím dochází k sestavení vnitřní sémantické struktury. Tato struktura je hierarchicky organizována a je paralelní k dlouhodobé sémantické znalosti. To tvoří základ pro doplnění sémantických detailů nižší úrovně. Po vyřešení problému v rámci vnitřní sémantické struktury může programátor napsat program v jazyce, který umí. To lze ale pouze za předpokladu shodné sémantické struktury jazyků.



Obrázek 3 Syntakticko-sémantický model řešení programovací úlohy žákem

Reisner (Reisner, 1981) založil svůj model na teoretickém principu procesu, podle kterého probíhá psaní dotazů. Tím je fungování procedurálních operací aplikovaných na strukturální uspořádání tabulek. Podle Reisnerova názoru uživatel v principu pracuje s myšlenkovými šablonami dotazu. Myšlenková reprezentace problému je z pohledu práce s přirozeným jazykem překlad slova do odpovídajících konstruktů v dotazovacím jazyce. Výsledek je potom zakomponován do syntaktické šablony dotazu. Na principu výše popsané teorie identifikoval Reisner několik typů transformací kognitivního překladu z přirozeného jazyka

do dotazovacího:

1) chybějící slovo v přirozeném jazyce uživatel nahradí klíčovým slovem konstruktu v dotazovacím jazyce,
2) uživatel odstraní slovo z reprezentace dotazu v přirozeném jazyce, neboť v dotazovacím jazyce není nutné,
3) výraz v přirozeném jazyce je nahrazen jiným výrazem dotazovacího jazyka.

Podle svého modelu Reisner navrhnul index „transformační složitosti“, který slouží k předpovědi složitosti dotazu, a tím i úspěšnosti transformačního procesu. Navrhnul výpočet indexu transformační složitosti zahrnující počet dílčích transformací z přirozeného jazyka do dotazovacího jazyka. Vysvětlil tím odlišné frekvence nesprávných odpovědí u transformace dvou různých vět z přirozeného jazyka do jednoho shodného dotazu. Vysokoúrovňový princip psaní dotazu objasnil na základě uživatelských strategií. Rozlišuje tři fáze psaní programu: formulace, plánování a kódování.
Mayer (Mayer, 1979) zkoumal rozdělení syntaktické úrovně poznání programovacího jazyka BASIC. Strukturně shodné prvky syntaktické znalosti seskupil do následujících úrovní:

1) strojová úroveň,
2) transakční úroveň,
3) předpříkazová úroveň,
4) příkazová úroveň,
5) povinná část programu,
6) základní nepovinná část programu,
7) vyšší část programu,
8) program.

Mayer sémantickou strukturu programovacího jazyka považoval za stěžejní koncept při formulaci obsahu a struktury výuky programovacích jazyků. Strojovou úroveň položil na pomyslnou nejnižší úroveň sémantické struktury jazyka BASIC. Ta je pevně spjata s hardwarem počítače a jedná se o úroveň elektromagnetických polí, logických obvodů a osmičkového kódu. Transakční úroveň popisuje obecné funkce počítače. Ty patří mezi základní konstrukční bloky příkazů. V terminologii programovacích jazyků je transakce jednotka znalosti programování a skládá se z operace, objektu a umístění. Příkladem

transakce může být „přesuň číslo, které je na vrcholu vstupního zásobníku, na výstupní zásobník“ (OP: MOVE, OB: číslo, LOC: input stack).

Každý řádek zdrojového kódu se může skládat ze série transakcí. Mayer se domníval, že popis příkazu v transakční terminologii usnadní porozumění vnitřních procesů počítače. Předpříkazy tvoří samostatnou podmnožinu kategorie příkazů a sdružují všechny příkazy obsahující stejné klíčové slovo. Příkazová úroveň je běžně nejnižší sémantická úroveň vysvětlovaná při výuce programovacího jazyka. Povinná část programu je složena z příkazů, které se musí vyskytovat společně. Základní nepovinná část programu je složena ze skupiny příkazů spojující účel použití. Vyšší nepovinné celky jsou vyšší celky sousledu příkazů. Může se jednat o sousled příkazů: načti proměnnou, vytiskni proměnnou a konec programu. Program je nejvyšší sémantickou úrovní a zastřešuje všechny výše uvedené sémantické úrovně programu. Skládá se z programových celků a příkazů. Analogicky lze uvažovat o vrstvách syntaktické znalosti dotazovacího jazyka. Ze syntaktického a sémantického rozboru struktury programovacího jazyka lze stanovit elementární úroveň znalostní úrovně nezbytné pro zvládnutí základů databázového myšlení.

Ve svém dalším výzkumu Mayer (Mayer, 1981) zkoumal sémantickou strukturu kompetencí pro řešení problému a své závěry ověřoval na procesu učení matematických úloh. Struktura sémantické znalosti je formována procesem učení a její charakter lze ovlivňovat použitou vyučovací metodou. Vyučování zaměřené na procvičování rychle aplikovat nově získané znalosti rozvíjí schopnosti související s propojením nových znalostí mezi sebou, tzv. vnitřní propojení. Zlepšení schopností řešení problémů lze docílit pomocí tzv. vnějšího propojení. To je specifické propojením nových znalostí s již existující sémantickou strukturou. Mayer ve svém článku položil fundamentální teorii pro strukturální rozbor obsahu výuky programovacích jazyků. Z ní potom vycházejí strukturální diagramy obsahu programovacích jazyků. Současně do kognitivního modelu učení programovacího jazyka zahrnul popis, jakým způsobem je v něm využívána krátkodobá a dlouhodobá paměť.

Jarke (Jarke & Vassiliou, 1985) navrhnul model dotazování, neboť jej potřeboval při evaluaci rozboru možností dotazovacího jazyka. Svůj model založil na vyjádření závislosti růstu produktivity na míře vstupní motivace a množství času stráveného přemýšlením nebo odlaďováním chyb. Růst produktivity ve svém modelu vyjádřil jako složenou proměnnou determinující kvalitativní a kvantitativní znaky řešení úlohy. Dílčí proměnnou, která může mít vliv na růst produktivity, definoval pomocí počtu redukcí myšlenkových kroků nutných k vyřešení problému. Vyšší nároky na zapamatování syntaktických konstruktů vedou k vyšší intenzitě myšlení. V rámci výzkumu dotazovacích jazyků se Jarke zabýval osobními

charakteristikami uživatele. Ty klasifikoval podle jejich interakčních schopností (syntaktické znalosti), znalosti programových konceptů a frekvence používání systému. Druhou úroveň klasifikace uživatelských typů založil na struktuře úlohy (sémantické znalosti), rozsahu operací a aplikační znalosti.

Ogden (Ogden, 1986) vytvořil třífázový kognitivní model psaní dotazu. Dotaz se podle něj formuluje v první fázi modelu. Při jeho formulaci uživatel vybírá relevantní data. Formulace může znít: „Kteří studenti získali v loňském roce prospěchové stipendium?“ Pro fázi formulace dotazu je důležitá pouze znalost aplikační domény. Druhá fáze na základě výsledku první fáze vybírá z datového modelu relevantní elementy a operace. Výsledek druhé fáze může být: „Ve výsledku dotazu musí být zahrnut sloupec jméno z tabulky student a ve sloupci stipendium.“ Ve třetí fázi student zadává již přeložený dotaz do počítače ve formě dotazovacího jazyka.

Schlanger (Schlager & Ogden, 1986) se věnoval vlivu vstupní úrovně a struktury znalostí učitelů při primárním střetu s výukou dotazovacího jazyka na výsledek učení. Při strukturálním rozboru koncipoval dva experimenty. V prvním experimentu sestavil neformální kognitivní model dotazovacího jazyka, jenž byl výsledkem kvalitativní analýzy strukturovaných rozhovorů se specialisty. Výsledný model zahrnul do experimentální dokumentace. Poté podrobil dvě skupiny žáků výuce dotazovacího jazyka a srovnal je dotazníkovým šetřením. Analýzu orientoval na vyhodnocení strukturálních rozdílů výsledku učení. Ve druhém experimentu použil kognitivní model pro popis konceptuální a procedurální informace obsažené ve studijních materiálech. Výsledek učení porovnal srovnávacím testem znalostí dotazovacího jazyka. Dva hlavní faktory determinující výkonnost při dotazování jsou podle jeho experimentů typ dotazu a konzistentnost kognitivního modelu žáka. Výše uvedený výzkum může použit jako metodika pro stanovení obsahu učiva elementárního střetu s databázovým myšlením.

Mayer (Mayer & Bayman, 1988) při výuce programovacího jazyka prováděl rozbor struktury výsledků učení s ohledem na obsah výuky. Za determinantu odlišnosti výsledků učení považoval studijní materiály. Hlavní důraz kladl na syntaktické a sémantické elémy obsahu učiva. U žáků podrobených výuce s důrazem na sémantickou složku jazyka bylo možné pozorovat menší četnosti výskytu nesprávného úsudku. Zároveň dosahovali lepších výsledků v testech orientovaných na řešení problému. Výkon při řešení problému byl tak v těsné souvislosti s konceptuální znalostí studentů. Výsledek experimentu Mayer považoval za důkaz rozvoje syntaktické, konceptuální a strategické znalosti učením programovacího jazyka. V kontextu disertační práce lze jeho závěr považovat za motivační klíč při zahrnutí

databázového myšlení do obsahu výuky.

Chan (Chan et al., 1993) se věnoval vlivu úrovně abstraktního myšlení na výkon žáka při rozvoji strukturovaného databázového myšlení. Standardní přístup klasifikuje datové modely do fyzické, logické a konceptuální úrovně. Na podobné bázi může být klasifikována interakce uživatele s databázovým systémem. Konceptuální úroveň v tomto pojetí zahrnuje výměnu informací s databází o okolním prostředí uživatele. Patří tedy princip odrazu reálných předmětů do databázového vyjádření v podobě entit, relací a atributech. Do logické úrovně databázového myšlení patří interakce uživatele a je založena na znalosti základních konceptů databázového myšlení, jako jsou relace nebo operace spojení. Znalostní báze logické úrovně vychází z paralelního položení ke konceptuální úrovni. Chan se experimentálně pokusil ověřit formulovanou hypotézu, zdali je konceptuální úroveň snazší na pochopení než logická. Pro ověření hypotézy Chan navrhnul experiment, ve kterém sledoval vliv znalosti ER modelu jako základu znalosti konceptuálního základu v porovnání s relačním modelem vyjadřujícím znalost logické úrovně. Výsledky potvrdily, že na konceptuální úrovni uživatelé dosahovali lepších výsledků v přesnosti práce a zároveň měli i větší jistotu. Na základě výsledku ověřené hypotézy se v kontextu disertační práce domnívám, že zahrnutí konceptuální úrovně do obsahu výuky databázového myšlení může poskytnout kvalitativně lepší výsledky učení než zahrnutí prvků rozvíjejících znalost na úrovni relačního modelu.

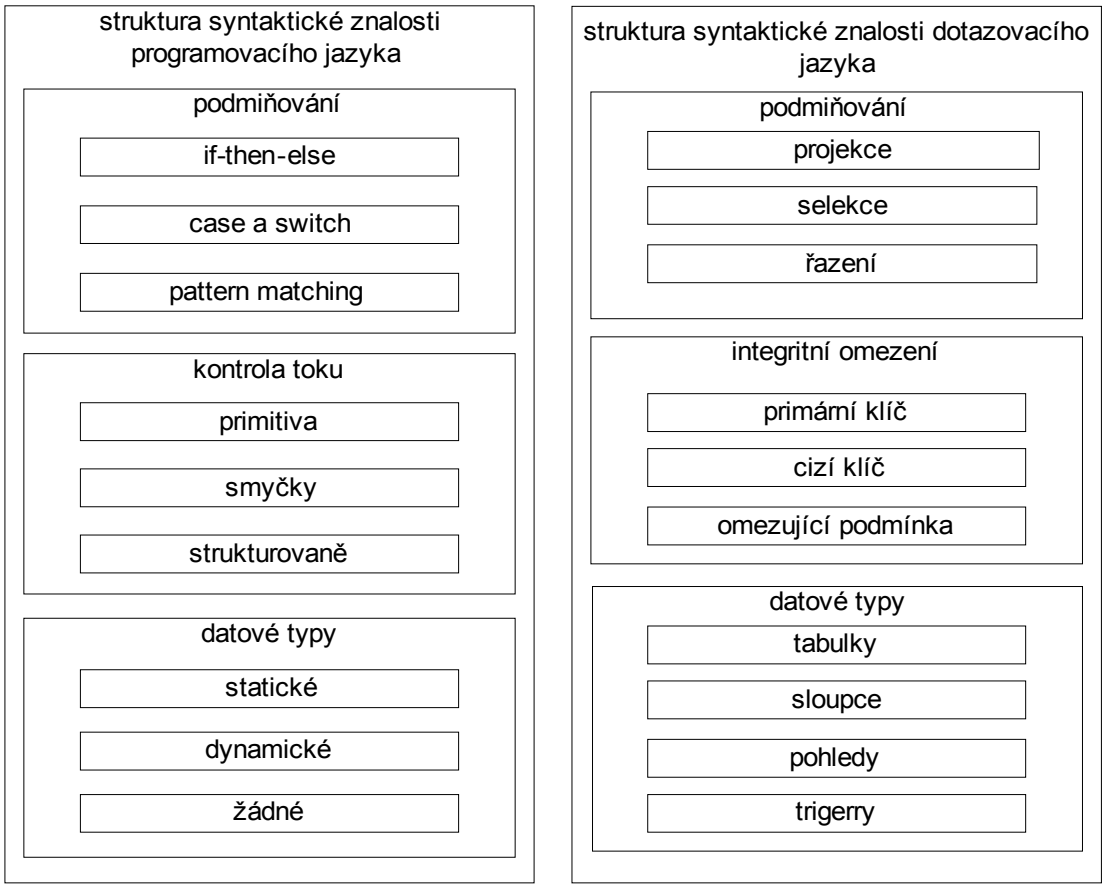
Třístupňový model později publikovaný Chanem (Chan et al., 1997) se orientoval na fázové rozpracování formulace dotazu. Za klíčový předpoklad pro korektní výběr potřebných dat považoval využití znalosti aplikační domény. Náplní druhé fáze byl proces výběru elementů z datového modelu a uplatnění znalostí použitých operací. Ve třetí fázi dochází k překladu výstupu z předchozích fází do dotazovacího jazyka. Popsaný model považuji za důležitou teorii s potenciální aplikací při rozdělení struktury obsahu výuky do skupin v kontextu procesní diverzifikace.

Prior (Prior, 2003) identifikoval v procesu formulace dotazu čtyři základní kroky:

1) výběr databázové tabulky obsahující požadovaná data,
2) definice způsobů, jak tabulky kombinovat nebo spojit,
3) v souladu se vstupními požadavky stanovit selekční kritéria pro výběr dat,
4) rozhodnutí o metodách pro zpracování tabulky.

Egan (Egan & Greeno, 1973) porovnával aplikaci dvou různých vyučovacích metod. První metoda byla založena na použití pravidel. Druhá byla založena na rozvoji činností

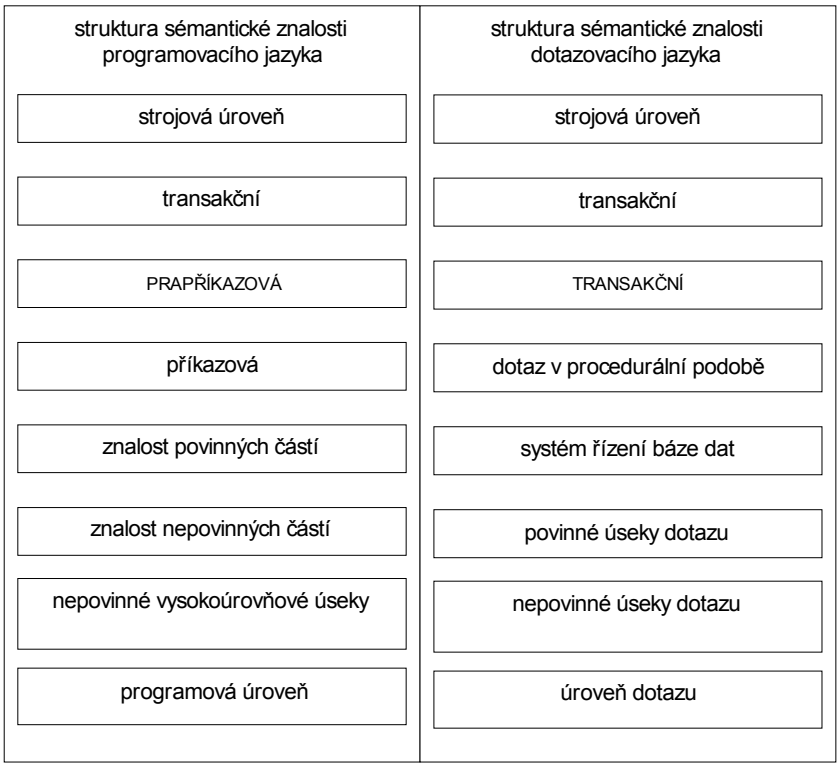
souvisejících s řízeným objevováním nových poznatků. Egan procesní komponenty obou vyučovacích metod srovnal, popsal rozdíl výsledku učení a vyvodil z nich optimalizační pravidla. Teoretické výsledky experimentálně ověřil ve výuce pravděpodobnosti aplikací různých vyučovacích metod. Podle jeho názoru se vyučovací metody odlišují v požadavcích kladených na znalosti a schopnosti studentů. Na základě výsledků experimentu formuloval hypotézu popisující strukturální rozdíl výsledků učení. Domníval se, že ve výsledku učení pomocí obou metod existuje rozdíl. Zatímco výsledek výuky metodou objevování je strukturální integrace již známých vědomostí, tak výukou pomocí pravidel může vyučující přidat nové poznatky do stávající struktury.



Obrázek 4 Komparace struktury syntaktické znalosti programovacího a dotazovacího jazyka

Kognitivní modely sestavování dotazu se odlišují v přístupu, úrovni podrobností a metodě ověření. Mayerovy (Mayer, 1981) modely se lišily počtem fází sestavování dotazu. Reisner (Reisner, 1981) a Ogden (Ogden, 1986) publikovali procesně orientované modely, jejichž základem bylo pořadí použitých kognitivních operací pro práci s datovou strukturou. Konstrukce modelu založeného na rozboru struktury dotazovacího jazyka zvolil Shneiderman

(Shneiderman et al., 1977). V modelu tak oddělil výsledek učení syntaktické a sémantické části dotazovacího jazyka. Svůj model založil na rozlišení syntaktické a sémantické složky dotazovacího jazyka stejně jako Mayer (Mayer, 1979). Jarke (Jarke & Vassiliou, 1985) pro svůj model provedl rozbor faktorů ovlivňujících výsledek učení a na jejich základě dotazovací jazyky klasifikoval. Analýzu souslednosti kognitivních operací při vymýšlení dotazu provedl Mayer (Mayer, 1981). Chan (Chan et al., 1997) se pokoušel o optimalizaci pořadí kognitivních operací při formulaci dotazů. Kombinací přístupů orientovaných na jednotlivé kognitivní operace v určitém pořadí ve spojení s myšlenkou struktury dotazovacího jazyka svůj model popsal Prior (Prior, 2003).



Obrázek 5 Komparace struktury sémantické znalosti programovacího a dotazovacího jazyka

Při konstrukci schematického procesu formulace dotazu v dotazovacím jazyce vycházím z práce o klasifikaci schémat pro organizaci znalostí. Tu publikoval Boyce (Boyce & Chamberlin, 1974) a definoval v ní tři hlavní kategorie schémat:

1) Deklarativní schéma.
2) Procedurální schéma.
3) Strukturální schéma.

Deklarativní schéma znalostí obsahuje informaci o tom, jak věci pracují a proč se chovají tak, jak se chovají. Dále zahrnuje jejich jména a pozice. Kromě toho obsahují informace o konceptech, elementech a vazbách mezi sebou navzájem v souvislosti s konkrétními subjekty.

Procedurální schéma znalostí obsahuje informace o podrobnostech kroků nebo aktivit, které jsou zapotřebí k dokončení úlohy nebo práce. Jedná se o typ znalostí, jejž lze převést do automatických, pravidelných procesů a u nichž lze definovat opakováním, které dovolují provést úlohu bez nutnosti zásahu vnějšího vlivu. Strukturní schéma znalostí je charakterizováno reprezentací poznatků v deklarativní paměti (kategorií pojmů), sémantických sítí, schémat zaměřených na řešení úlohy, scénářů, reprezentace obsahů v procedurální paměti (produkční systém), v rámci experimentů směřujících k nalezení optimálního vlivu výukových metod na podporu smysluplného učení se vědci zabývali konstrukcí kognitivního modelu procesu učení počítačového jazyka. Výše uvedené kategorizace schémat organizace znalostí jsem použil při systematizaci rozvoje databázového myšlení. Deklarativní schéma jsem v oblasti databázového myšlení vyjádřil jako obecný model databázového myšlení. Procedurální schéma jsem navrhnul jako procesní model formulace dotazu. Strukturální schéma bylo podkladem pro systematizaci databázového myšlení.

Konkrétní modely myšlení v oblasti programování se zaměřením na znalost dotazovacího jazyka publikoval Mayer (Mayer, 1975). Ty mohou sloužit jako podklad pro rozpracování teorie výuky databázového myšlení, zaměřený primárně na zjištění výsledku učení dotazovacího jazyka nebo programovacího jazyka obecně. Ve svém dalším výzkumu se Mayer (Mayer, 1981) zabývá otázkou sestavování dotazu v kontextu interakce s okolím a práce s pamětí. Mayer (Mayer, 1979) kromě rozdělení znalostí na syntaktickou a sémantickou v rámci obou skupin provedl horizontální strukturaci znalosti podle její složitosti. Tento poznatek lze využít v rozpracování kognitivního modelu znalosti dotazovacího jazyka. Pro sémantickou znalost Mayer (Mayer, 1981) deklaroval jako nezbytné především její postupné učení v kontextu již existujících znalostí a zdůraznil nezbytnou důkladnost takového přístupu. Jiný Mayerův (Mayer & Bayman, 1988) výzkum nalezl odlišnost výsledků učení v souvislosti s použitými studijními materiály.

Do stejné oblasti výzkumu programování patří Shneidermanův kognitivní model (Shneiderman, 1977). Jeho základní přínos lze spatřovat především v rozdělení znalosti dotazovacího jazyka na sémantickou a syntaktickou. Implikace tohoto rozdělení se odrážejí

i v jeho kognitivním modelu znalosti dotazovacího jazyka. Jeho poznatky lze v kontextu disertační práce aplikovat při strukturaci učiva. Dále je možné zjištění aplikovat do oblasti výuky databázového myšlení. Učivo je možné rozčlenit na související se sémantickou a syntaktickou znalostí. V oblasti databázového myšlení lze tedy v rámci sémantických principů vyučovat obecnější principy, jako je rozšiřování slovní zásoby, principy informačních zdrojů, principy úložiště informací, obecné principy fungování dotazovacích jazyků. V oblasti syntaktického přístupu je potom již vyučován konkrétní produkt na vytváření dotazníků, dotazovací jazyk nebo program vytvářející sestavy.

3 Obsah rozvoje databázového myšlení

Jak vyplynulo z předchozí kapitoly, bylo při bližším pohledu na horizontální vrstvení obsahové dimenze databázových systémů z pohledu stupně jejich osvojení identifikováno šest úrovní. První vrstvu nazývám propedeutická, druhou nácvičná, třetí úroveň expansivní, čtvrtou stabilizační, pátou integrační a šestou aplikační. Rozvrstvení kompetenční dimenze vychází z úrovně znalostí, kterou má student. Každá vyšší vrstva přepokládá znalost nižší vrstvy. Při bližším pohledu na kompetenční dimenzi kontextu databázového myšlení lze za nejdůležitější rovinu označit propedeutickou úroveň. Domněnka vyplývá z analýzy důsledků komplexity tématu databázových systémů, jenž se projevuje nároky na výuku databází. Domnívám se proto, že je nezbytné, aby předpoklady k němu byly rozvíjeny již na úrovni primárního vzdělávání a pozornost byla soustředěna především na rovinu propedeutickou, neboli průpravnou.

Při analýze databázového myšlení z pohledu jeho obsahu je mým hlavním cílem nalézt stabilní prvky databázového myšlení. Jedná se o komponentu databázového uvažování, jež je v čase a napříč produkty relativně nebo maximálně neměnná. Může mít podobu principu fungování, úseku pracovního postupu, sémantické komponenty nebo syntaktického konstruktu databázového jazyka. Hlavní metoda pro detekci stabilního prvku je založena na analýze chronologie vývoje databázového systému v čase nebo napříč produkty a destilaci shodných prvků. Předpokládám, že aplikací výše uvedeného postupu lze dospět ke stanovení kvalitních podkladů pro tvorbu učiva vedoucího k rozvoji databázového myšlení.

Cíle základů databázového myšlení analyzuji na základě poznatků o struktuře učiva databází a v jejich rámci se zaměřit na kritické a náročné oblasti. Ty mohou být potenciálním zdrojem komplikací při praktické práci s databází. Po jejich identifikaci si kladu za cíl nalézt způsob jejich rozvoje na propedeutické úrovni. Jejich návrh může být základem pro stanovení metod měření výsledků učení databázového myšlení.

3.1 Vývoj databází

V kontextu poznání cílů výuky databázového myšlení považuji za nezbytné se na prvním místě poohlédnout za historickým vývojem databázových systémů. Dle mého názoru je to nezbytný předpoklad pro správné určení cílů výuky databázového myšlení. Zároveň exkurz do vývoje datových bází poskytuje velice důležitý základ pro odhad budoucího směru rozvoje databází.

Na počátku vzniku databázových systémů byla myšlenka zkonstruovat deduktivní program. Jedná se o zvláštní kategorii programu, jejž lze považovat za předchůdce dnešních databázových systémů. Jejich první implementace vznikaly na počátku šedesátých let dvacátého století. Jednalo se o nejranější pokusy počítačem zpracovat přirozený jazyk. Běžná implementace deduktivního programu po položení otázky prohledala bázi faktických údajů a vybrala nejvhodnější odpověď. Jednou z prvních implementací deduktivních programů byl DEDUCOM (DEDUCTIVE COMMUNICATOR). Slagle (Slagle, 1965) uvádí kapacitu 68 faktických údajů a schopnost odpovědět na 10 položených otázek. DEDUCOM používal heuristický algoritmus.

DEDUCOM byl jeden z mnoha podobných programů v té době. Rozvoj deduktivních programů směřoval ke zvyšování výkonu celého systému především navýšením kvantity faktických údajů. Koncept se však stal základem pro návrh programu, který by sám o sobě byl schopen psát jednoduché programy (umožňoval při hledání odpovědi na otázku sestavit vyhledávací program). Při jeho vývoji se však objevily komplikace spojené s principem implementace hledání odpovědí. Na některé otázky nelze najít odpověď a na jiné pouze v případě, kdy jsou potřebná fakta seřazena ve správném pořadí. Program DEDUCOM problém hledání řešil matematickým aparátem výrokové logiky. To ale přinášelo nové implementační obtíže, neboť fakta musela být před vložením do systému DEDUCOM transformována do logického ekvivalentu ve výrokové logice, a proto bylo nutné zadat mnoho dalších redundantních údajů. Ale i tak se DEDUCOM stal důkazem konceptu programu, jenž umožňuje simulované zodpovězení otázky.

Podrobný přehled a hodnocení systémů pro zpracování otázky v přirozeném jazyce publikoval Simmons (Simmons, 1965). Uvedl seznam celkem patnáct systémů. Ty rozdělil do tří skupin podle vnitřního uspořádání údajů. Simmons si při porovnávání systémů položil otázky související s vývojem systémů zpracovávajících přirozený jazyk. Identifikoval problematiku zpracování víceznačných slov, překlad z přirozeného do formálního jazyka a návrh algoritmu prohledávající rozsáhlé stromové datové struktury. Dále se věnoval pohledu do historie počítačových systémů určených k hledání odpovědí na otázky v přirozeném jazyce. Z jeho článku lze sledovat linii jejich vývoje až do roku 1959. Základním principem fungování podobných programů je princip překladu zadaných vět z přirozeného jazyka do symbolů. Shodným znakem zůstává nutnost překladem vytvořené symboly zpracovat pomocí aparátem formální logiky.

Jedna z potenciálních implementací algoritmu pro zpracování symbolů je systém založený na implementaci datové báze. Ta byla v původních deduktivních systémech reprezentována

strukturou seznamu symbolů. Příkladem první implementace na principu datové báze byl systém SAD SAM. Ten četl jednoduché věty v angličtině a získával z nich atomické údaje. Po analýze přečtených údajů je zařadil do své znalostní báze. Primární určení programu SAD SAM byla jazyková analýza věty. Systém měl za úkol určit ve větě podstatné jméno, sloveso, přídavné jméno a ostatní slovní druhy. Určování slovního druhu bylo v systému realizováno algoritmem provádějícím větný rozbor. Větný rozbor byl implementačně založen na principu transformace větných členů do stromové struktury. Ta zahrnovala nejenom druh každého slova, ale i vztahy mezi jednotlivými větnými členy. Výstup větného rozboru byl vstupem pro sémantický analyzátor, který hledal shodu přísudku s podmětem, a tak určoval slovní druhy.

Ve druhé polovině šedesátých let dvacátého století vzniká na základě zkušeností s implementací deduktivních systémů a vnitřního uspořádání jejich datových bází teorie relačních databází. Příchod relačního konceptu umožnilo rozpracování obecné teorie informačních systémů.

Z předchozího vývoje vyplynula především nutnost vždy znovu implementovat prohledávací algoritmy, definice vnitřního uspořádání faktů a implementace výrokové logiky. Jako jeden z mnoha vyvíjel obdobný obecný informační systém i Bryant (Bryant & Semple, 1966). Jeho implementace se ale od ostatních systémů odlišovala především precizním oddělením algoritmů pro správu datové báze od ostatních částí systémů. Tím Bryant vnesl do oblasti obecných informačních systémů myšlenku oddělení implementační části informačního systému v podobě samostatného podsystému řízení báze dat.

Další projekt z doby, kdy vznikala teorie relačních databází, se nazýval RAND. Podle Leviena (Levien & Maron, 1967) byla hlavním úkolem projektu RAND logická analýza velkých objemů faktických údajů. Projekt se účastnil posílení vývoje a urychlení implementace relačního datového souboru. V této oblasti byla jeho hlavním přínosem realizace myšlenky strukturování datového souboru a posun poznání v trendu logické analýzy velkého souboru dat. Jeho datový soubor měl na počátku 200 000 datových vět. Ve srovnání s DECUCOM to je 3000krát větší objem informací. Ty byly uloženy v prototypu relačního informačního jazyka. V tom bylo možné komunikovat s počítačem a zadat programu problém pro logickou analýzu. Největší implementační problémy se nalézaly v oblasti implementace prototypu relačního informačního jazyka, návrhu logické analýzy programovacího jazyka a organizací vnitřního uspořádání úložiště dat. V pokročilé fázi implementace obsahovala datová báze systému RAND 105 nebo 106 datových vět. Pro spolehlivé uspokojení požadavků logické analýzy faktických dat byla komunikace se systémem možná pomocí jazyka založeného na predikátové logice, protože umožňoval dobře popsat data podle jejich

vlastností a vnitřního uspořádání relací mezi nimi.

3.2 Databázová terminologie

V rámci celé disertační práce používám termíny, které patří do teorie databázových systémů. Mým cílem není definovat je zde přesnými, matematickými definicemi, ale popsat je s cílem nabídnout vhled do základů databázové terminologie. Popis se tak stane východiskem pro jejich pochopení s cílem analyzovat databázové myšlení v kontextu školního vzdělávání.

Pojem databázový systém ve své disertační práci používám pro množinu údajů s vnitřní vazební strukturou a množinou programů určených pro přístup k nim. Za hlavní oblast využití databázových systémů považuji pracovní prostředí vyžadující získávání, organizaci a ukládání informací. Jádro databáze, systém řízení báze dat (SŘBD), je navržen za účelem správy velkého objemu informací, datových vět, a zároveň je připraven na jejich nárůst a změnu struktury. Systém řízení báze dat umožňuje definici specifické struktury datové báze s ohledem na typ ukládané informace a poskytuje mechanismus umožňující manipulaci s ní. Pokud mají být data sdílena více uživateli, systém musí předejít možným anomáliím způsobeným současným přístupem více uživatelů ke stejným datům. Databázový systém tak umožňuje uživateli abstraktní pohled na data realizovaný zapouzdřením konkrétních podrobností o způsobu uložení a správě dat. Data v datové bázi mohou být uložena ve složitých datových strukturách, ale i přesto systém řízení báze dat zprostředkovává efektivní rozhraní pro přístup k nim.

Typický model databáze poskytuje tři hlavní pohledy na uložená data. Ta jsou seřazena od nejnižší po nejvyšší z hlediska abstrakce dat a nazývají se vrstva fyzická, logická a uživatelská. Na nejnižší úrovni abstrakce, fyzické úrovni, je popsán způsob uložení dat a podrobnosti o datové struktuře. Vyšší úroveň abstrakce, logická úroveň, popisuje, jaká data jsou uložena a jaký je vztah mezi nimi navzájem. Nejvyšší úroveň abstrakce, uživatelský pohled, popisuje část databáze, ke které může přímo přistupovat uživatel, a poskytuje pro každého uživatele vlastní pohled. Aplikační program je program využívající přístup k databázi a někdy je řazen do samostatné, aplikační vrstvy.

Pohled na strukturu databáze z uživatelské vrstvy poskytuje databázové schéma. Databázové schéma specifikuje vnitřní uspořádání struktury, sémantiku a omezující podmínky pro přístup k údajům. Entitně-relační (E-R) model zobrazuje strukturu databázového schématu. Na obecné úrovni je složen ze základních prvků, nazvaných entity, a zobrazuje jejich strukturu a vzájemné vazby mezi nimi.

Entita je obraz věci nebo objektu ze skutečného světa odlišitelného od jiného objektu. Entity

jsou popsány v databázi pomocí specifikace vyjadřující jejich charakter a mají množinu atributů sloužících k zachycení jejich vlastností. Relace vyjadřuje vztah mezi entitami. Základní princip získání informací z databáze spočívá v prohledání konkrétní relace. Pokud se v relaci vyskytuje příliš velký objem informací, je běžně implementován mechanismus umožňující rychlejší přístup k nim. V případě relačních databází je výše popsaný princip implementován na fyzické vrstvě použitím indexů.

Databázové schéma je definice množiny, vyjádřená v datovém definičním jazyce. Výsledek provedení příkazů v datovém definičním jazyce je množina informací uložených ve zvláštním souboru nazvaném datový slovník. Datový slovník obsahuje metadata. Ta lze popsat jako informace o datech uložených v databázi. Datový slovník používá systém řízení báze dat před tím, než začne manipulovat nebo definovat datovou strukturu databázového systému. Manipulace s daty je množina transakcí zahrnující získávání, vkládání, odstraňování a modifikaci informací uložených v datové bázi. Jazyk určený k manipulaci s daty dovoluje uživatelům přistupovat a manipulovat data ve shodě s jejich vnitřní strukturou.

Systém řízení báze dat se při práci s daty řídí podle datového modelu datové báze. Kromě jiných existují dva základní typy jazyků pro manipulaci s daty. Prvním zástupcem jsou procedurální jazyky pro manipulaci s daty. Při implementaci programu v procedurálním programovacím jazyce musí programátor specifikovat jednak popis dat, se kterými má program pracovat, a jednak definovat algoritmus metody jejich transformace. Na rozdíl od toho skupina neprocedurálních programovacích jazyků umožňuje manipulovat s daty bez toho, aniž by uživatel musel stanovit, se kterými daty chce pracovat. Kromě toho není nutné specifikovat, jaké transformace má algoritmus na datovou bázi aplikovat. Do skupiny neprocedurálních programovacích jazyků řadíme i dotazovací jazyky. V rámci dotazovacích jazyků je základním příkazem, neboli větou pro komunikaci s počítačem, dotaz. Hlavním smyslem zadávání dotazů databázovému systému je manipulace s vnitřní strukturou datové báze.

Jak jsem již uvedl, manipulace s databází prostřednictvím systému řízení báze dat má dva hlavní účely neboli případy použití. Relační dotazovací jazyk obecně obsahuje jednak podskupinu dotazů pro definici dat (Data Definition Statement, DDL), a jednak podskupinu dotazů pro manipulaci s daty (Data Manipulation Language, DML). I když již bylo navrženo mnoho dialektů databázových jazyků vycházejících z podobného principu, dnešním průmyslovým standardem v oblasti dotazovacích jazyků je strukturovaný dotazovací jazyk (Structured Query Language, SQL). Ten je implementován v převažující většině všech databázových produktů na trhu a stal se de facto průmyslovým standardem.

3.3 Konceptuální model

Základem abstraktního vzoru konceptuálního modelu pro manipulaci se strukturou relační databáze je původně myšlenka účelné organizace prvků informačních systémů. Dle mého názoru je nejcitovanějším autorem v oblasti relačního modelu určeného pro velké sdílené banky dat E. F. Codd (Codd, 1970). Když publikoval svůj článek zabývající se matematickým aparátem sloužícím jako základ implementace relačního modelu, byl zaměstnán ve výzkumné laboratoři IBM, ve městě San Jose ve státu Kalifornie. Hledal metodu, jak uživatele izolovat od organizace dat v počítači obdobně, jak to tehdy dovolovaly deduktivní programy. Formuloval tedy požadavek, aby byly zachovány pracovní postupy uživatele, i když se změní vnitřní struktura datové báze. Tím umožnil oddělit pracovní postupy uživatele pro manipulaci se strukturou od vnitřního uspořádání datové báze.

Codd vycházel z předpokladu dynamické variability vnitřního uspořádání databáze vlivem časté aktualizací dat, modifikace výstupních sestav a přirozeného nárůstu množství informací uložených v databázi. Oddělením procesu zpracování požadavku uživatele manipulujícího se strukturou od strojové reprezentace vnitřních mechanismů Codd dosáhl zachování pracovních postupů s datovou bází. Separaci struktury informací od uživatelského rozhraní navrhnul implementovat ve formě dotazovacího jazyka. Ten neměl brát ohled na pořadí jednotlivých prvků v rámci datové struktury na fyzické úrovni databáze, zavedl princip umožňující rychlejší přístup k větším množstvím dat a specifikoval způsob abstrakce přístupové cesty k datům v bázi.

Codd (Codd, 1970) se zabýval i budoucím směrem rozvoje dotazovacího jazyka. Pro základní funkční aparát jazyka použil predikátovou logiku. Hlavní požadavek, jak by měl správně fungovat dotazovací jazyk, jej vedl k myšlence spojit argument s uloženou relací i v případě, že relace není známa, ale existuje. Dále definoval jako hlavní vlastnost dotazovacího jazyka jeho schopnost zapouzdřit se v jiných hostitelských jazycích. Základním principem univerzálnosti datového jazyka se tak stala jeho popisová schopnost. Vychází tak najevo charakteristický rys flexibility v podobě schopnosti ve strukturovaném dotazovacím jazyce formulovat nejenom datovou strukturu, ale pomocí stejných jazykových konstruktů s ní manipulovat. V tom spočívá hlavní rozdíl mezi neprocedurálním dotazovacím jazykem a procedurálními jazyky zaměřenými na popis algoritmu postupné transformace elementů systému.

Ve svém pozdějším článku (Codd, 1981) obhajuje Codd relační databáze z hlediska produktivity. Na počátku osmdesátých let dvacátého století se výrazně zvýšila poptávka po nových aplikacích a relační databáze mohly řešit dva zásadní problémy. První spočíval

v poptávce po přímém kontaktu s informacemi uloženými v počítačích. Druhý problém byl spojen se zvýšením produktivity programátorů při vývoji aplikačních programů.
Datový model je spojen s datovou strukturou modelu, operacemi a omezením operací. Jedna z jeho klasifikací je založena na úrovni abstrakce. Dvě hlavní úrovně abstrakce jsou logická a konceptuální úroveň. Na logické úrovni je uživatelská znalost logické struktury, existence klíčů a cizích klíčů a operací, jako je spojení a omezení k reorganizaci datové struktury. Na logické úrovni je ER model a logická struktura dat, jako jsou klíče, cizí klíče a operace typu spojení a omezení. Na konceptuální úrovni je hlavní ER model, který je široce používán pro konceptuální modelování.

3.4 Dotazovací jazyk

Za programovací jazyk obecně považuje Khosrow-Pour (Khosrow-Pour, 2005) způsob zápisu počítačového algoritmu. Aby počítač vykonal zadanou úlohu, je nutné mu zadat sekvenci instrukcí, jiným slovem algoritmus. Pokud je posloupnost příkazů vyjádřena v programovacím jazyce, nazývá se program.
V současné architektuře počítačů je srdcem celého počítače procesor: ten vykonává instrukce v takzvaném strojovém jazyce. Vyjádřit složitější algoritmus ve strojovém jazyce je však obtížné nejenom pro programátory. Je tomu tak kvůli principu činnosti procesoru. Ten dokáže pracovat pouze s binární číselnou soustavou a v ní provádět několik základní operací. Instrukce ve strojovém jazyce se tak skládá pouze z operace a hodnoty binárně zapsané do paměti počítače. Mezi myšlením člověka a strojovým jazykem tak vzniká sémantická mezera, jejímž řešením je zavedení právě programovacích jazyků. Těch existuje široká škála a liší se především svojí komplexitou, která se úměrně zvyšuje tím, jak se přibližují přirozenému uvažování člověka. Smyslem vyšších programovacích jazyků je nabídnout programátorovi co nejsrozumitelnější a nejsilnější programovací prostředí. Toho vyšší programovací jazyky dosahují implementací složitějších konstruktů a různými abstrakcemi datových struktur. Způsob, jakým je možné, aby počítač rozuměl vyšším programovacím jazykům, je existence takzvaného překladače, který tlumočí program v programovacím jazyce do strojového kódu.
Typicky se programovací jazyk skládá ze tří základních konstruktů. První je jazykový konstrukt sloužící k deklaraci a definici dat. S jeho pomocí může být vytvořena vytvořená datová struktura programu. Druhým konstruktem programovacího jazyka je knihovna funkcí a procedur. Ty mohou být jednak dodány s programovacím jazykem, jednak je možné si vytvořit vlastní. Třetím konstruktem je potom vlastní přepis algoritmu do programovacího jazyka jako sekvence příkazů a ty potom řídí tok programu.

Formální jazyk je složen z abecedy nebo slovníku a obsahuje konečnou množinu symbolů. Slovo je složené ze symbolů abecedy o konečné velikosti. Věta je složena ze slov. Jazyk je jakákoliv množina vět. Gramatika slouží k přesné definici syntaktické správnosti sestavené věty v určitém jazyce. Kromě toho gramatika poskytuje strukturální popis vět. Formální gramatika a formální jazyk vycházejí z pokusu popsat abstraktní principy jazykové struktury anglického jazyka, jenž by byl použitelný pro napsání programu pro umožňující jazykovou analýzu. (Hopcroft & Ullman, 1969)

Základem konceptuálního jádra dotazovacího jazyka je formální jazyk založený na aparátu formální gramatiky. Gramatika formálního jazyka je v kontextu dotazovacího jazyka ucelená množina pravidel popisující princip, jakým lze složit ze slabik slova a ze slov věty. Lingvistika zná pojem generativní gramatika jako strukturovanou množinu základních elementů, na které aplikuje pravidla. Aplikací pravidel na strukturu elementů generuje gramatika nové datové prvky. V rámci generativní gramatiky existují dva základní principy jejich vytváření. První princip postupuje od kategorizace vět, na které jsou aplikována pravidla platná pro daný úsek věty. V další fázi jsou pravidla aplikována na úroveň slov. Druhý princip spočívá v opačném postupu, tzn. odspoda nahoru. Implementace druhé metody je frekventovanější v oblasti kategoriální gramatiky a výchozím bodem je použití lexikálních prvků, nazvaných generátory. Na generátory se v další fázi aplikují pravidla, čímž vytvoří bázi. Na jejím základě se podle gramatických pravidel budují komplexní struktury. Jiný pohled na gramatiku nabízí kombinace logiky a algebry. Na jejím základě lze následně specifikovat vlastnosti množiny. (Wilson & Keil, 2001)

Dotazovací jazyk určený pro komunikaci s datovou bází je formální neprocedurální jazyk založený na principu formální gramatiky. Právě v rámci dotazovacích jazyků je teoretický koncept formálního jazyka aplikován jako mechanismus zabezpečující oblast syntaktické správnosti dotazů.

Přehledem neprocedurálních jazyků, jejich kategorizací, evaluací a vzájemným srovnáváním se ve svých výzkumech zabývalo mnoho různých vědců (Welty & Stemple, 1981), (Leavenoworth & Sammet, 1974), (Turner et al., 1985). Leavenoworth podává komplexní přehled o neprocedurálních jazycích, stanovuje jejich základní charakteristiky a definuje problémy skupiny neprocedurálních programovacích jazyků. Obecně je, jak jsem již uvedl, program definován jako přepis řešení daného problému. Procedura je posloupnost uspořádaná sekvence kroků. Základní charakteristika procedurálního programování je nutnost přizpůsobit se architektuře a podstatě sekvenční organizace konvenčního (Von Neumannova) digitálního počítače. Základní princip, který odlišuje neprocedurální programování od procedurálního,

spočívá v potenciálu umožňujícím sestavit předpis řešení problému bez ohledu na sekvenční uspořádání. V širším pojetí je neprocedurální program předpisem pro řešení problému bez ohledu na detaily toho, jak má být řešen.
Aby bylo možné neprocedurální jazyky kategorizovat, vytvořil Leavenoworth metriky neboli kvantitativní znaky. Ty sloužily zároveň jako základ pro vzájemné srovnávání neprocedurálních programovacích jazyků. Metriky sestavené za účelem porovnávání neprocedurálních programovacích jazyků jsou například následující:

1) Počet příkazů v daném jazyce a počet příkazů, které mohou být vyjádřeny pomocí jiných příkazů, a tedy nejsou základní.
2) Množství informací o aplikační doméně, které musí být specifikovány v daném jazyce.
3) Množství sekvencí nebo paralelismů, které jazyk dovoluje.

Leavenoworth dále uvedl základní typické charakteristiky specifické pro neprocedurální jazyky. Prvním charakteristickým rysem je asociativní odkazování. Jedná se o takovou vlastnost neprocedurálního jazyka, jenž umožňuje vyjádřit odkaz na prvek bez explicitního vyjádření adresní cesty nebo bez určení postupu, jakým má být prováděno prohledávání specifických datových struktur. Popis principu asociativního odkazování uvedl ve svém článku již Codd (Codd, 1970). Ten princip asociativního odkazování považoval za základní element definice relačních operací a domníval se, že jeho implementací bude navýšena relativní měřitelná síla jazyka bez ohledu na typ datové struktury.
Kromě tradiční množiny operací kartézského produktu (spojení, průniku) definoval Codd specifické relační operace typu projekce, spojení, rozdělení a omezení. Definované relační operátory poskytují různé typy asociativního odkazování. Kromě principu asociativního odkazování jsou další charakteristickou vlastností neprocedurálních jazyků agregační operátory. Agregační operátory jsou základním principem, který při psaní umožňuje programu vyhnout se konstruktu známému z procedurálních jazyků jako iterace. Codd za tím účelem použil koncept algebraických operátorů, který aplikoval na relace, a tím vznikl princip agregačních operátorů. Eliminace explicitního použití iteračních sekvencí je tedy jedním z dalších příznačných rysů neprocedurálních prvků v programovacím jazyce.
Elementární konstrukt agregačního operátoru lze ale využít i při implementaci zobecněného algoritmu pro seskupování a vytváření součtů v datové bázi. Tak je možné do neprocedurálního jazyka vložit funkční schopnost zvládnout jedny ze základních úloh

hromadného zpracování dat. Princip slučovacího operátoru je funkce vybudovaná nad uspořádanou relační množinou a umožňuje pro každý prvek kartézského produktu vybrat pouze entity, které vyhovují zadané podmínce. Ta může být specifikována jako ekvivalent hodnot v různých doménách. Třetím znakem neprocedurálních jazyků je přísné oddělení vnitřní struktury údajů z pohledu řazení nebo uspořádání ve vztahu k aplikaci. Ve funkčním programu je implementací tohoto principu eliminována datová závislost a explicitně určená struktura programu. Kromě výše uvedených, dle mého názoru nepodstatnějších, existují i další neprocedurální vlastnosti. Mezi ně patří nedeterministické programování a paralelismus.

V historickém vývoji dotazovacího jazyka se ne vždy jevilo jako naprosto jisté, zda má být dotazovací jazyk procedurální, nebo neprocedurální. Výzkumem rozdílu mezi procedurálním a neprocedurálním dotazovacím jazykem se zabýval Welty (Welty & Stemple, 1981). Ve svém výzkumu srovnával dva dotazovací jazyky SQL a TABLET. Zaměřil se na potenciál obou jazyků sestavit složitý dotaz. Evaluaci provedl na základě experimentálního výzkumu ekvivalentních vlastností zkoumaných jazyků. Oba jazyky jsou v principu založeny na teoretickém konceptu relačního modelu a jejich komplexita je přibližně ekvivalentní.

Jako hlavní cíl svého výzkumu si stanovil srovnat odpovídající konstrukty obou jazyků. Dílčím cílem, který vyplýval z hlavního cíle, bylo navrhnout jejich zlepšení. Welty dospěl k názoru, že při úloze porovnávání počítačových jazyků jsou klíčovým zdrojem evaluace jejich parametrů lidé. Ti mohou přispět nejenom k ověření, zdali existuje možnost se jazyk naučit, ale zároveň umožňují včasnou eliminaci budoucích potíží spojených již v raných fázích návrhu a vývoje programovacího jazyka. Z experimentu vyplynulo, že pokud programátor použije procedurální programovací jazyk, může lépe psát složité dotazy než programátor, který píše program v programovacím jazyce obsahujícím méně procedurálních rysů.

Postupný vývoj programovacích jazyků směřoval k implementaci dotazovacích jazyků s větším množstvím neprocedurálních jazykových konstruktů. Předchůdcem dnes nejrozšířenějšího neprocedurálního dotazovacího jazyka SQL byl SEQUEL. Boyce (Boyce & Chamberlin, 1974) ve svém článku o SEQUELu popisuje teoretický základ, podle kterého byl rozpracován implementační koncept počítačového jazyka určeného pro přístup k datům v integrované relační datové bázi. Hlavní funkcionalita jazyka SEQUEL (Structured English Query Language) vycházela z množiny jednoduchých operací predikátové logiky aplikovaných na datovou strukturu tabulky. Syntax neprocedurálního dotazovacího jazyka SEQUEL tvořila konzistentní množina klíčových slov v anglickém jazyce. Jednalo se

o množinu příkazů vycházejících ze znalosti pracovních postupů lidí s tabulkami. Hlavní pozornost byla věnována pracovním postupům pro získávání informací z tabulky.

SEQUEL umožňoval spojovat základní prvky do složitějších dotazů. Byl určen nejenom pro profesionální programátory, ale i pro odborníky nebo experty v dalších oborech. Vývoj jazyka tak reagoval na požadavek transformace procedurálních specifikací k deklarativním specifikacím problému. Jedním z hlavních důvodů, proč SEQUEL implementoval podporu deklarativní specifikaci problému, byl pokus o snížení nákladů na vytvoření, správu a modifikaci programu napsaného v tomto jazyce. Hlavní klíč k úspěchu konceptu strukturovaného programování bylo zjednodušení a zrychlení programování a to znamenalo i nižší cenu softwaru. Dalším klíčovým faktorem úspěchu byla vzrůstající potřeba pomoci neprofesionálním uživatelům k efektivní komunikaci s datovou bází.

Hlavním cílem snažení v první polovině sedmdesátých let proto bylo vyvinout způsob, jakým by uživatelé mohli sami zvládnout komunikaci s počítači. Řešení mělo být co nejblíže přirozenému jazyku a zároveň zahrnovalo požadavek na uskutečnění co nejjednodušší interakce se systémem pomocí výběru z připravených nabídek programu. SEQUEL byl ale primárně určen pro skupinu uživatelů ochotných naučit se vysokoúrovňový, neprocedurální programovací jazyk. Při práci s neprocedurálním dotazovacím jazykem SEQUEL uživatel zadává proměnné obsahující stejné hodnoty, jako jsou řádky nebo části řádků relací. Dotaz je sestavený na základě booleovské algebry a obsahuje univerzální nebo existenciální kvantifikátory. SEQUEL používá základní terminologii, jako je tabulka, doména (sloupec tabulky), rozsah a argument. Množina hodnot v rozsahu sloupce pojmenované tabulky musí odpovídat argumentu. Hlavní přínos jazyka SEQUEL spočívá v zavedení návrhové metody shora-dolů. To je hlavní myšlenkový proces, který je hojně využíván při psaní základního bloku dotazovacího jazyka skládáním elementů SELECT-FROM-WHERE. Při psaní dotazu následuje doplnění elementární kognitivní struktury o název sloupce a tabulky ze struktury datové báze (např. sloupec PŘÍJMENÍ z tabulky ZAMĚSTNANEC). Implementace výše popsaného principu vede k redukci komplexity výrazových prostředků dotazovacího jazyka na elementární stavební bloky a množinu pravidel pro jejich vzájemnou kombinaci.

Okrajově se zmíním ještě o dalším neprocedurálním dotazovacím jazyku, kterým byl SQUARE (Specifying Queries As Relational Expressions). Primárně byl určen pro vyjádření základních dotazů (přístup, modifikace, vkládání, mazání) nad datovou bází a k tomu využíval charakteristiku relací vztaženou k jejich změnám v čase. To, co jej odlišovalo od ostatních příbuzných jazyků, byla jeho relační úplnost (Boyce et al., 1974), (Boyce et al., 1975). Relačně úplný jazyk umožňuje, aby jím byl vyjádřen libovolný výraz predikátového

počtu. Jeho vývoj byl stejně jako SEQUEL založen na abstrakci pracovních postupů lidí s tabulkami.

Jak uvádí Reisner (Reisner, 1981), vědci začali ve druhé polovině sedmdesátých let dvacátého století provádět experimenty s databázovými dotazovacími jazyky, jejichž hlavním cílem byla orientace na koncové uživatele. Společným jmenovatelem podobných experimentů bylo spojení poznatků behaviorálních a informačních věd. Aby bylo možné zjednodušit používání dotazovacích jazyků a přiblížit je uživateli, bylo nutné definovat evaluační kritéria. Pro účely kvantitativního měření tak byla například stanovena definice pojmu jednoduchost používání. Ze stanovených evaluačních kritérií vyplynuly měřicí techniky, jež umožnily zkoumat chování uživatelů při vytváření aplikací pro různé vědecké projekty. Reisner ve svém pokusu srovnával použitelnost dvou dotazovacích jazyků SQUARE (Boyce et al., 1974), (Boyce et al., 1975) a SEQUEL (Boyce & Chamberlin, 1974). Ve svém článku popisuje Reisner obtíže související s návrhem formálního dotazovacího jazyka. Kromě toho je v článku uveden i podrobný popis metodologie, podle které byl experiment veden. Skupinu studentů nechal Reisner řešit úlohy zaměřené na psaní, čtení a interpretaci dotazů, porozumění otázce, učení zpaměti a sledoval u nich kompetence k řešení problému. Ty se potom staly základem pro výzkum chování studentů a výsledky experimentu byly publikovány s ohledem na teorii konstrukce dotazovacího jazyka.

Jako další se dotazovacím jazykům věnoval Jarke (Jarke & Vassiliou, 1985). Jeho jméno uvádím pro specifický přístup založený na myšlence sledování kvality jazyka podle poměru ceny a výkonu. V souladu s tím lze vyvinout metodiku pro evaluaci dotazovacího jazyka, což umožňuje jednoduše kvantifikovat výsledky porovnávání. V rámci výzkumu vyjádřil Jarke cenu jako počáteční náklady na zaučení a náklady jako výdaje na používání jazyka. K výkonnostní charakteristice dospěl kvantifikací funkčních konstruktů jazyka, jako je například výsledek výběru a skládání, výstupní prezentace a flexibilita interakce.

Z výše uvedeného je patrné, že dotazovací jazyk, jenž se podobá dnes používanému, vznikal již v 80. letech 20. století. To je důvod, proč dotazovací jazyk považuji za relativně stabilní element databázového myšlení.

3.5 European Computer Driving License

Při hledání obsahu výuky databází, jež by tvořily kompetenční jádro databázového myšlení, vycházím z existujících standardů. První standard, který jsem podrobil bližší obsahové analýze, je databázový modul European Computer Driving License (ECDL). V souvislosti s rozvojem databázového myšlení je v rámci ECDL relevantní databázový modul. Cílem

rozvoje kompetencí podle ECDL je rozvoj praktické práce s databázovým prostředím. Zahrnuje nejenom praktické porozumění principu fungování databázových systémů, ale i rozvoj kompetencí v oblastech návrhu databázové struktury, tvorby a používání aplikační vrstvy datové báze. Výukový cíl látky databázového modulu integrální součástí konceptuálního cíle ECDL. Tím je rozvoj informační gramotnosti občanů. Poskytuje jim ve volitelném rozsahu možnost ověřit své znalosti a externí agenturou potvrdit, zdali jejich znalosti dosáhly ve vybraných oblastech informační gramotnosti požadované úrovně. A právě z požadavků stanovených standardem vycházím při hledání výukových cílů databázového myšlení. Vzhledem k zaměření standardu, to jest evaluovat úroveň znalostí informačně gramotného občana, koresponduje cíl ECDL i se zamýšlenou konstrukcí výukových cílů databázového myšlení. V kontextu disertační práce se tak právě databázový modul ECDL jeví jako vhodný kandidát pro stanovení kompetenčního jádra databázového myšlení na úrovni znalostí studenta střední školy. Zároveň slouží jako vodítko pro stanovení celkové koncepce tematického celku databází. Podrobný obsah ECDL je uveden v příloze 10.3 Databázové kompetence dle ECDL.

V databázovém modulu kompetenčního modelu ECDL je stanoven rozsah kompetencí, jejichž znalost musí uchazeč prokázat. Souvisí jednak s teoretickým pochopením principu fungování databázových systémů a za druhé se schopností ji používat. Cílovým stavem kompetencí v rámci modulu ECDL je sice zvládnutí konkrétního návrhového prostředí databází, ale charakteristika cílových dovedností odpovídá obecným požadavkům na principiální zvládnutí praktické práce s databázovým systémem. Základním znalostním předpokladem umožňujícím zvládnutí průpravy pro práci s databázovým systémem je ale v prvé řadě znalost teoretických konceptů pro práci s ní. Výše uvedený fakt reflektuje i obsah databázového modulu kompetenčního modelu ECDL. Jedná se o základní premisu bezproblémového zvládnutí praktické části práce s databázovým produktem, neboť toho nelze dosáhnout bez potřebného porozumění terminologie a vzájemných souvislostí odborných pojmů. Dalším klíčovým prvkem vedoucím ke zvládnutí databázového myšlení je korektní myšlenková reprezentace struktury základních strukturních elementů datové báze. V případě relačních databází se jedná o schopnost představit si strukturu tabulek a vazeb mezi nimi. Ke kognitivní reprezentaci statických prvků vnitřního uspořádání datové báze se váže znalost dynamiky a vazeb pracovních postupů pro manipulaci s nimi. To souvisí s oblastí návrhu, definice a specifikace struktury základních strukturních elementů.

Pokud hovoříme o relačních databázích, stejně jako kompetenční model ECDL, je základním strukturním elementem tabulka databáze. Znalost principu vytváření tabulky v databázi je dle

mého názoru jedním z hlavních předpokladů pro zvládnutí návrhu databáze. Zvládnutí pracovního postupu pro vytvoření a úpravy tabulky jsou jedním ze základních požadavků databázového modulu ECDL. To v sobě zahrnuje i postup pro aplikaci úprav strukturálního uspořádání struktury datové báze, jako je formulace datových typů polí tabulky a jejich korekce. Po dokončení fáze návrhu databáze je však zapotřebí do existující struktury datové báze umět záznamy nejenom vkládat, ale i provádět jejich změny a odstraňování. Na úrovni návrhu aplikační vrstvy je ve vztahu k databázi základní dovedností příprava uživatelského rozhraní v souladu s datovou strukturou tabulky. V případě návrhového rozvržení a definice funkcionalit formuláře je stěžejní úloha prostředku jako uživatelsky přívětivého prostředku pro manipulaci s údaji v datové bázi. Je zároveň základem pro zpřístupnění operací pro řazení a filtraci dat uživateli. V rámci kompetenčního modelu ECDL je zahrnuta i schopnost upravovat a spouštět databázové dotazy přímo z vybraného návrhového prostředí. Další znalostní oblastí je schopnost rozlišovat relevanci a důležitost údajů uložených v datové bázi. V rámci této oblasti je nutné znát princip fungování mechanismu dotazu jako teoretický základ pro správnou definici omezujících podmínek s ohledem na očekávaný výsledek požadavku zadaný databázovému stroji. V souladu se základními znalostmi souvisejícími s vytvářením výstupů z databáze je kromě toho nutné ovládat i schopnost vytvoření výstupní sestavy anebo dovednost definovat výstupy z databáze tak, aby mohly sloužit k další distribuci a zpracování.

Standard ověřuje i úroveň znalosti terminologie relačních databází a v rámci kompetenčního modelu stanovuje konkrétní pracovní postupy, jejichž znalost musí uchazeč o certifikát prokázat. V rámci terminologické oblasti jsou zahrnuty termíny vyskytující se v popisech principu fungování databázových systémů a deskripcích pracovních postupů pro strukturaci údajů v databázi.

Klíčové pojmy v rámci kompetenčního modelu databázového modulu ECDL jsou databáze, tabulka, primární klíč, index, pohled, formulář, data, informace, záznam, pole, dotaz, sestava, řazení, filtr, vyhledávání, kritéria dotazu a zástupní znaky.

Databázový modul kompetenčního modelu ECDL pojímá terminologii obecně. V praktické části ECDL zkoušky je však nezbytné prokázat požadované kompetence nejenom na úrovni porozumění jednotlivým termínům a teoretickým konceptům, ale i na úrovni praktického zvládnutí pracovních postupů formou plnění zadaných úkolů v časovém limitu. V kontextu teoretické části kompetenčního modelu ECDL je stěžejní porozumění rozdílu mezi daty a informacemi.

Výstižné vysvětlení poskytuje Mudrák (Mudrák, 2007), který ve své disertační práci téma

rozebírá v rámci formulace obsahu informační gramotnosti. Tu definuje jako provázanou soustavu schopností a dovedností jedince. Z jeho pohledu je zacházení s daty v jejich různých podobách a formátech, vyhledávání dat v dostupných informačních zdrojích, definování kritérií vyhledávání dat a používání softwarových nástrojů pro hromadné zpracování dat základní součástí práci s nimi. Při prezentaci nalezených dat je podle jeho definice klíčové posouzení jejich významu s ohledem na jejich aplikaci pro řešení konkrétních problémů. Dále je podle něj důležité hodnocení jejich důvěryhodnosti a schopnost těžit a čerpat z nich v dané chvíli relevantní a korektní informace.

Výše uvedené činnosti zahrnuje do procesu interpretace dat. Výsledkem procesu interpretace dat jsou informace. Pro informační gramotnost je klíčová kompetence schopnost manipulace se strukturou informace, její třídění, organizace dle zadaných kritérií a znalost principu jejich vzájemných souvislostí. Zvládnout interpretaci informací současně s jejich efektivním použitím v procesu učení, transformaci nabytých informací a jejich poznání ve smyslu kvalitativních a kvantitativních změn individuálních poznatkových struktur spadá v Mudrákově modelu informační gramotnosti do oblasti učení. Při nabývání znalostí jsou podle něj důležité specifické metody práce s informacemi, považování si znalostí jako osobního vlastnictví a kapitálu, reflexe vlastních poznávacích procesů. Nové a modifikované poznatky lze znovu kódovat do vhodného formátu dat: jejich prezentace a věcná argumentace ve prospěch svých postojů, efektivní zaznamenávání znalostí ve formě znalostních či poznatkových map, komunikace a kooperace v týmu patří do oblasti kódování znalostí a je východiskem pro tvorbu dat. Tolik Mudrák k rozdílu mezi daty a informacemi.

Základním principem, jejž je dle mého názoru zde nutné zmínit, je postup pro správné formování databázového myšlení na základě správné představy jednotlivých vrstev databázového systému. Ten v principu odděluje fyzickou vrstvu od logické a aplikační vrstvy. Při pokusu o hlubší pochopení principu fungování aplikační vrstvy je však nezbytné rozumět i principu fungování a vzájemných propojení s logickou a fyzickou vrstvou databázového systému. To zahrnuje obsáhnutí specifických informací vázaných ke konkrétním rolím, jež člověk může při rozvoji databázového myšlení zastávat. Správný vhled na princip součinnosti vrstev datové báze je základním předpokladem pro korektní klasifikaci a kategorizaci principů, pracovních postupů a evaluaci funkčních možností nástrojů konceptuálního modelování. V neposlední řadě je v rámci aplikační vrstvy zásadní znalost databázového dotazovacího jazyka, který umožňuje nad logickou vrstvou dat formulovat požadavky vedoucí k získání požadovaných údajů.

Na úrovni fyzické vrstvy strukturního uspořádání datové báze jsou v kontextu relačních

databázových systémů základní pojmy tabulka, záznam a pole. Stručně lze jejich princip a součinnost specifikovat následovně. Tabulka má údaje zaznamenány v jednotlivých řádkách, které se skládají ze sloupců. Každý sloupec tabulky má svůj datový typ. Ten určuje charakter údajů, které je možné do sloupce tabulky vložit. Buňka je jedna položka záznamu ve sloupci určitého datového typu.

Databázový modul ECDL předpokládá kromě znalosti principů fungování databázového systému i znalost souvislostí mezi praktickou aplikací rozsáhlých databází a uživatelským pohledem na jejich funkčnost. Rozsáhlé databáze jsou například rezervační systémy leteckých společností, databáze státní správy, informační systémy bank a nemocniční systémy pro evidenci pacientů. Je dobré si uvědomit, že pokud si například objednáváme letenku v rezervačním systému letecké společnosti, vede k jeho dokončení většinou pracovní postup zahrnující vyplnění formuláře. Z něj jsou údaje přenášeny po síti do datové báze, která záznam o naší rezervaci uloží do tabulky rezervací. Zpětné vyhledání záznamu informace o rezervaci letenky je v principu dotaz do tabulky záznamů o letovém rozvrhu spojený se záznamy z tabulky obsahující údaje o pasažérech.

Z pohledu konstrukčních principů je struktura databáze v některých ohledech specifická a odlišná od jiných informačních struktur. Konkrétně databázový modul ECDL v oblasti návrhu databáze požaduje znalost principu datové normalizace. Jedná se o soubor pravidel určujících způsob specifikace typu, formátu a podmínek. Na jejich základě se řídí rozdělení záznamů do tabulek. Ty tvoří základ datové normalizace databáze. Aby nedocházelo k redundantním výskytům údajů v databázi, nebo naopak nevznikl výkonnostní problém ukládání záznamů do tabulky, je možné na logické úrovni specifikovat rozdělení záznamů. To může být stanoveno podle účelu jejich použití a vyplývá ve vhodné oddělení jednotlivých atributů a optimální definici primárních klíčů. Cílem datové normalizace je v souladu se zaměřením datové báze vhodně definovat její strukturu. Rozhodnutí o aplikaci datové normalizace vychází z rozhodnutí o účelu struktury záznamů a musí být v souladu s předpokládaným zaměřením využití databáze.

Obecně lze konstatovat, že datová normalizace je metoda pro optimální strukturaci datové báze. Jiné požadavky na datovou normalizaci budou vyplývat pro systémy určené ke zpracování velkého množství dat v reálném čase a jiné pro systémy využité v oblasti podpory rozhodování. Dalším úkolem definice vnitřního uspořádání databáze je stanovit principy zajištění datové integrity. Tu je možné na úrovni datové báze vynutit pomocí tzv. omezujících podmínek. Pro všechny údaje zaznamenávané do databáze je vhodné stanovit hodnoty, kterých mohou nabývat. Při jejich specifikaci se vychází z definice jejich datového typu. To

umožňuje stanovit pravidla ověřování vkládaných údajů. Z nich potom může vyjít logika určená k rozhodování, které údaje budou do tabulky zaznamenávány.

V souvislosti s výše uvedeným principem je nutné připomenout potřebu specifikovat datové typy. Aby struktura databáze korespondovala se zamýšleným druhem vkládaných údajů, lze stanovit velikost a typ datového typu ukládaného údaje. Datový typ může být například text, číslo, datum nebo čas. Každý datový typ může být svázán podrobnější definicí omezujících podmínek. Tím může být zajištěno ověřování správnosti a formy údaje na úrovni databázového stroje před jeho uložením do datové báze. Může tak dojít k ověření velikosti vkládaného záznamu, vynucení formátu ukládaného údaje a v případě, že nebyl specifikován žádný údaj, vložení předdefinované hodnoty. Korektní definici výše uvedených omezujících údajů lze použít i při návrhu a vytváření vstupního formuláře. Tím lze například zajistit, že formulář před uložením záznamu bude obsahovat všechny požadované údaje, nebo zajistit, že vkládané údaje budou mít vyžadovaný formát anebo budou splňovat nároky stanovené zadanou podmínkou.

Stěžejním konceptem pro zajištění identifikace jednotlivých údajů v datové bázi je implementace jedinečnosti každého vkládaného záznamu. Princip unikátnosti může být implementován specifikováním jedinečné charakteristiky každého vloženého řádku. To je v relačních databázových systémech obvykle implementováno pomocí možnosti specifikace primárního klíče tabulky. Primární klíč ke každému záznamu přidá jedinečný identifikátor, podle kterého lze zajistit unikátní přístup k jednotlivým záznamům. Zároveň implementace primárního klíče slouží v relačních databázových systémech jako jedinečný identifikátor pro spojení údajů z více tabulek. Aby relační datová báze poskytovala rychlý přístup ke konkrétnímu údaji uloženému v datové bázi, implementuje strukturní prvek jménem index. Jeho princip spočívá v uložení informace o pozici záznamu. Lze jej použít při vyhledávání konkrétního záznamu a jeho vhodné nasazení lze sledovat jako rychlejší přístup k požadovanému záznamu databázového systému.

V relačním databázovém systému lze modelovat logickou souvislost entit ve formě definice relací mezi tabulkami. Prakticky hlavní účel relací je minimalizace výskytu redundantních dat. Metodou, která umožní jeho implementaci, je datová normalizace. Ta eliminuje redundantní data úpravou struktury tabulek aplikováním pravidel takovým způsobem, aby bylo možné uvést redundantní údaj pouze jednou nebo s minimálním počtem opakování. Pokud existuje stejná kombinace souvisejících údajů a lze je vyjádřit relací mezi více tabulkami, existuje možnost spojit je referenční vazbou primárních klíčů. Tím lze docílit výrazného omezení paměťové náročnosti na uložení dat a kromě toho zajistit minimalizaci

výskytu redundantních dat. Principiální základ relace spočívá v aplikaci primárních klíčů do role vazebních prvků mezi tabulkami. K tomu, aby byla zajištěna integrita referenčních vazeb primárních klíčů, je nezbytné aplikovat pravidla principu referenční integrity. Ta slouží k udržování integritních, tedy neporušených a validních relací mezi tabulkami. S referenční integritou databáze souvisí i princip pro jejich udržování a správu. Ten je založen na implementaci principu kaskádového mazání záznamů a metody pro validaci vkládaných záznamů ověřením integrity vazeb relací. Pro případ, že do jedné z tabulek vkládáme údaj a ta je v relaci s jinou tabulkou, dochází před vložením takového záznamu ke kontrole referenční integrity. To znamená, že systém řízení báze dat ověří, zdali opravdu existuje související záznam a zda vkládaný záznam na něj správně odkazuje. Pokud systém řízení báze dat odpovídající záznam nenalezne, na základě specifikovaných pravidel vyvolá akci, které může ve výsledku zabránit vložení nového záznamu. Výše uvedený mechanismus tak zajistí nejenom konzistenci údajů, ale může zabránit i vložení nevalidního údaje. Tím systém řízení báze dat zabrání vložení záznamu bez správné vazby na jinou tabulku v souladu s definovanou relací. Zároveň je tak vynuceno dodržení referenční integrity.

V případě, že by bylo při zobrazování tabulek použito spojení více tabulek, jejich korektní spojení zajistí právě definice referenčního spojení přes primární klíč. To je základní princip spojování tabulek a výsledkem je volba správného korespondujícího záznamu z jiné tabulky. Kromě principu aplikace referenční integrity pro vkládání záznamu existuje obdobný mechanismus pro zajištění správné aktualizace záznamů napříč několika tabulkami. Tento druh referenční integrity je založen na dodržení pravidla aktualizace záznamů na základě stejného principu jako při jejich vkládání. Pokud je zadán požadavek na změnu záznamů v tabulkách spojených relací, systém řízení báze dat provede kontrolu jejich aktualizace tak, aby nebyla porušena vazba na jinou tabulku. Pokud by dotaz požadoval aktualizaci v rozporu s definovaným integritním omezením, systém řízení báze dat transakci nepovolí a nedojde k její realizaci. Dalším případem aplikace referenční integrity může být implementace podmínek, na jejichž základě dochází k mazání záznamů z tabulek vázaných relací. V tom případě je možné referenční integritu vynutit pomocí aplikace kaskádového mazání záznamů. To je založeno na aplikaci principů relačních vazeb při odstraňování údajů z tabulek s ohledem na vzájemné vazby mezi nimi. Pokud je proveden pokus o odstranění záznamu z jedné tabulky a ten přitom odkazuje na další záznamy v jiných tabulkách, lze systému řízení báze dat zadat pravidlo, podle kterého provede mazání záznamů i v souvisejících tabulkách.

Pracovní postupy s databází lze klasifikovat podle jejich charakteru. Můžeme tak vytvořit kompetenční skupiny pracovních postupů specializovaných na konstrukci dotazů, strukturu

tabulek, rozvržení formulářů nebo návrh sestav. Výše uvedené členění klade důraz na úlohu databáze jako informačního zdroje dostupného i bez znalosti vnitřní implementace datové báze.

Předpoklad oddělení implementační vrstvy od uživatelské vyslovil již Codd (Codd, 1970). Položil tím základ rozvoje obecného konceptu datové báze. Ten uživateli umožňuje používat databázový systém, aniž by musel znát fyzickou vrstvu databáze. Obecně lze rozdělit práci s databází do dvou hlavních skupin. První skupinu tvoří odborná práce s databází zahrnující návrh vnitřního uspořádání, správu uložených údajů nebo návrh aplikačního rozhraní. Jedná se o kompetence, jež patří do oblasti dovedností databázového specialisty. Kromě nich je databázový specialista zodpovědný za regulaci přístupových práv k datové bázi, správu datové báze a schopnost obnovy správného fungování databáze při jejím nekorektním chování. Druhou skupinu tvoří převážně uživatelské dovednosti zaměřené na získání údajů z databáze.

Při praktickém používání návrhového prostředí existuje mnoho různých dovedností a pracovních postupů, které je nezbytné si osvojit. První ve výčtu činností, které je nutné zvládnout, aby bylo možné používat aplikaci pro návrh databáze, je její spuštění a ukončení. V rámci databázové aplikace je pak možnost modifikovat konkrétní implementaci databáze. Návrh je zahájen jejím otevřením a ukončen uložením a uzavřením databáze.

Další dovednost určená databázovým modulem kompetenčního modelu ECDL je vytvoření nového návrhu databáze a otevření a uložení rozpracovaného návrhu na konkrétní místo souborového systému operačního systému. V konkrétním návrhovém prostředí databázové aplikace, kterou popisuje databázový modul ECDL, je specifickou součástí práce s ním nástrojová lišta neboli panel nástrojů. Na něm jsou umístěny ovládací prvky pro návrh databázové aplikace, jež mají za úkol usnadnit orientaci v prostředcích návrhu databáze. Kromě ovládacích prvků databázová aplikace obsahuje též návod na použití, který se odborně nazývá programová nápověda.

Při běžné práci s databázovou aplikací je nutné zvládnout otevírání, ukládání a zavírání návrhových modelů tabulek, dotazů, formulářů a sestav. V rámci jednotlivých součástí aplikace pro návrh databáze je pro návrhové modely tabulek, dotazů, formulářů a sestav nezbytné zároveň znát i možné druhy jejich zobrazení. Tabulka tak může být zobrazena buď v návrhovém zobrazení umožňujícím pracovat se seznamem jmen sloupců tabulky, s jejich datovými typy, integritními omezeními a typem jedinečnosti záznamů, který je u nich povolen. Pro datovou tabulku, jak již bylo uvedeno výše, je možné specifikovat její jméno, typ a platná omezení pro vkládané údaje.

Na dotaz, formulář a sestavu lze pohlížet jednak v návrhovém pohledu. To je režim zobrazení umožňující specifikaci struktury rozložení a princip funkčnosti. Další návrhový zobrazovací mód tabulky souvisí přímo s konkrétním obsahem tabulky. Lze tak sledovat údaje, které jsou již do datové báze zadány. Mezi operace, které patří do běžného pracovního postupu návrhu databáze, patří kromě přidávání tabulek, dotazů, formulářů a sestav i jejich odstraňování ze strukturální báze. Pro každý druh zobrazených údajů, jako je tabulka, dotaz a formulář, je stanovena jedinečná sada ovládacích prvků, které umožňují pohyb po jednotlivých záznamech. Obecně se jedná o ovládací prvky, jež umožňují navigaci na první, poslední, následující, předcházející nebo konkrétní záznam tabulky. Specifikace přesunu na konkrétní hodnoty může být provedena například výběrem hodnoty primárního klíče záznamu. Záznamy v tabulce je ale možné řadit i podle jiných kritérií. To lze obecně specifikovat jako vzestupné nebo sestupné uspořádání vybraných prvků. Systém řízení báze dat umožňuje uspořádat záznamy tabulky buď podle číselného pořadí primárního klíče nebo na základě kritérií korespondujících s abecedním pořadím.

Práce s údaji v tabulce jsou v rámci databázového modulu kompetenčního modelu ECDL omezeny na dvě základní operace. První z nich je vytvoření nového záznamu v tabulce. Druhá operace je odstranění záznamu v tabulce. Kromě toho lze upravovat strukturu jednotlivých záznamů. To umožňuje pracovat s obsahem dat uložených v datové bázi. Obecně se tedy jedná o modifikaci existujících údajů datové báze. Jedná se o operace přidávání, mazání a úpravu dat v záznamech.

V oblasti administrace a návrhu relačního databázového systému je hlavní pozornost soustředěna na operace určené pro manipulaci se strukturou tabulky. Základní dovedností, jež tvoří nezbytný předpoklad pro administraci databází, je znalost pracovních postupů pro návrh, modifikaci a odstraňování tabulky. Pří návrhu tabulky se jedná konkrétně o schopnost tabulku navrhnout a pojmenovat. Dále specifikovat strukturu jednotlivých polí tabulky a schopnost definovat datový typ sloupců tabulky. Určování datového typu konkrétního atributu relační tabulky je založeno na stanovení charakteristiky vkládaných údajů. Je tedy nezbytné dobře specifikovat všechny možné hodnoty, kterých by mohlo dané pole nabývat. Kromě datového typu existují ještě další vlastnosti, které je možné poli přiřadit. Jedná se především o formát zobrazení údaje. Ten je založen na požadované charakteristice zobrazení údaje ve formuláři nebo v sestavě a odvíjí se od požadavku na jejich vzhled.

Mimo to lze specifikovat výchozí hodnotu datového pole. Tím je zajištěna hodnota, která bude zobrazena v poli pro případ, že by tabulka obsahovala prázdnou hodnotu. Jedná se o charakteristickou vlastnost využitelnou mimo jiné u primárního klíče tabulky. Díky

specifikaci pravidel pro vyplnění výchozí hodnoty pole tabulky je možné pomocí sekvenčního principu specifikovat inkrementální hodnotu vloženou pro případ založení nového řádku. Tím lze zajistit jedinečný identifikátor vloženého záznamu. Princip definice sekvence tak může například zajistit, aby každý další vložený řádek měl při svém záznamu do tabulky hodnotu primárního klíče o jedničku vyšší, než je hodnota primárního klíče záznamu předchozího pole.

Jak již bylo uvedeno výše, umožňuje databáze zajistit pravidla pro ověřování správnosti vkládaných údajů. Pro případ, že vznikne požadavek na bližší specifikaci vkládaného údaje, je možné použít mechanismy ověřující jeho korektnost. Do databáze lze implementovat zajištění korektních a konzistentních údajů. Při změně údajů může systém řízení báze dat dle stanovených pravidel ověřit, zdali změna vyhovuje definovaným podmínkám. Změnou může být vkládání, modifikace nebo mazání hodnot z datové báze. Na výše uvedeném principu lze implementovat ověřování počtu číslic, formátu vkládaného data, času nebo měny.

Dále existuje možnost specifikovat formát údaje, který musí být splněn při jeho zadání do formuláře tak, aby mu zadaný údaj vyhovoval, eventuelně byl podle něj korigován. Další oblastí, se kterou se můžeme setkat při návrhu databáze, je princip zachování struktury údajů vlivem modifikace datového typu a vlastností pole tabulky. Je nutné si uvědomit, že pokud změníme datový typ pole v tabulce, může dojít ke ztrátě dat nebo interpretaci odlišné od podoby, ve které byla data zadávána. V případě, že snížíme informační kapacitu sloupce tabulky, dojde k redukci informační kapacity záznamů. V praxi to znamená, že pokud máme tabulku se sloupcem, u kterého je specifikováno, že může pojmout řetězec obsahující dvacet znaků, a my provedeme redukci a zadáme maximální kapacitu sloupce na deset znaků, u všech záznamů s více než deseti znaky budou přebytečné znaky nenávratně z databáze odstraněny. Stejným principem můžeme odstranit desetinnou část čísla, pokud změníme specifikaci sloupce z reálného čísla na číslo celé.

Vlivem takové modifikace struktury datové báze lze ztratit přesnost údajů. Při redukci informační kapacity databázového pole je tedy nutné brát ohled i na data, která již databáze obsahuje. S dalším pravidlem pracovního postupu se setkáváme při specifikaci primárního klíče relační tabulky. Pokud bychom chtěli stanovit primární klíč u již existujícího sloupce tabulky, je nutné brát v potaz, že u takového sloupce musí být zajištěna jedinečnost údajů ve sloupci obsažených. Je tedy nezbytné, aby každé pole sloupce obsahovalo jedinečnou hodnotu.

Při konstrukci struktury databáze se činnostmi související s návrhem tabulky. Jedním z nich je přidání nového sloupce do existující tabulky. Před přidáním nového sloupce je nezbytné znát

datový typ sloupce a rozhodnutí, zdali požadujeme u vkládaných údajů vynucení omezující podmínky nebo formát zobrazení údaje. Při specifikaci zobrazení sloupce v tabulce je možné definovat vlastnosti spojené se šířkou jeho zobrazení. V případě, že je nezbytné specifikovat formát zobrazení údaje vybraného z konkrétního sloupce a ten nevyhovuje formátu zobrazovaných údajů, lze jeho zobrazení upřesnit přesně na míru zobrazovaným datům. Tím vzniká prostor k náhledu pouze na zobrazované údaje bez toho, aby údaj spotřeboval více místa na zobrazovacím zařízení, než je zapotřebí. Pomocí stejného principu lze korigovat interaktivní prvky pro zobrazování údajů, jako je například mechanismus pro výběr konkrétních hodnot z vložených údajů ve sloupci.

Základním principem pro získávání informací z databáze je aplikace omezujících podmínek. Tím lze v rámci tabulky vyhledat údaje, které omezující podmínku splňují. Z uživatelského pohledu se použití výše uvedeného principu používá při vyhledávání. Tím, že uživatel specifikuje vyhledávací kritéria, dochází k transformaci vzoru vyhledávání do kritéria výběrového dotazu. Je to tedy způsob, kterým je ze zadané podmínky sestaven dotaz. Ten umožňuje projít vybraný sloupec tabulky a každou hodnotu porovnat se zadanou hodnotou. Pokud vzoru zadaná hodnota vyhovuje, je výsledek zařazen do množiny všech údajů, které systém řízení vrátí jako výsledek dotazu. Při dalším kroku procesu výběru návratových hodnot dotazu se vybraná množina přeloží do aplikační vrstvy, kde je zobrazena na základě specifikovaných zobrazovacích kritérií. Jedná se o princip, jenž může být použit na aplikační vrstvě nejenom v samostatném vyhledávacím dialogu, ale i jako možnost aplikace filtru na již zobrazené údaje. Dvě základní metody při volbě filtrovacího kritéria jsou jednak specifikace zobrazovací podmínky, jednak v grafickém rozhraní může být implementováno filtrování výběrem zobrazené hodnoty. Princip stanovení omezujícího kritéria může být implementován nejenom při zobrazení tabulky, ale i v případě zobrazení formuláře. Pokud při zobrazení tabulky na aplikační vrstvě specifikujeme filtr, dojde pouze k výběru hodnot, které dané podmínce vyhovují. Existuje však i opačný proces pro zobrazení všech záznamů z tabulky a tím je odstranění omezující podmínky.

Stěžejním tématem pro oblast databázového myšlení je oblast dotazovacího jazyka. Formulace dotazů je základním předpokladem pro zvládnutí práce s databází. Jedná se komunikační prvek, jehož základ je shodný pro většinou databázových produktů na trhu.

Dotazovací jazyk slouží nejenom k pokládání dotazů pro vyhledávání údajů v databázi. Kromě toho může být použit v oblasti změn struktury databáze, modifikace vnitřního uspořádání datové báze anebo řízení přístupu k databázovému systému. Kromě toho některé dialekty dotazovacího jazyka nabízejí možnost specifikovat analytické výrazy, pomocí

kterých je možné vytvářet složité pohledy na údaje uložené v datové bázi nebo pracovat s fyzickým uspořádáním databázové struktury na úrovni operačního systému. Při požadavku na specifikaci omezující podmínky, jejímž účelem je vybrat pouze některé údaje z datové báze, je optimálním prostředkem použití dotazu. Správným postupem pro formulaci je stanovení omezujícího kritéria. Podle něj budou odpovídající údaje porovnány a případné shody zařazeny do množiny výsledků hledání. Dotazovací jazyk nabízí kromě vyhledávání v jedné tabulce konstrukty prohledávající relace tabulek. Po spojení údajů z několika tabulek dle existujících primárních a cizích klíčů, jsou definována omezující kritéria výběru ze spojených tabulek, určena pravidla pro seskupení údajů a nastaven druh uspořádání výstupních údajů. Výše uvedený postup zahrnuje hlavní kritéria, která umožňuje většina implementací dotazovacích jazyků. Při definici výběrových kritérií jsou hlavními prostředky specifikace dotazu operátory relační algebry. V případě číselných údajů tak můžeme používat například operátory rovná se, nerovná se, menší než, větší než, menší než nebo rovno, větší než nebo rovno. V případě textových údajů dovoluje dotazovací jazyk implementovat možnost použití funkčních operátorů, jako je výběr podřetězce textového pole, vyhledání znaku v rámci textového pole, stanovení počtu znaků textového pole anebo definice podmínky výběru záznamu porovnáním se specifikovaným řetězcem. Typický příklad výše uvedeného případu použití může být specifikace výrazu pro porovnávání určitého rozsahu písmen. Kromě specifikace jednotlivých kritérií existuje možnost kombinovat podmínky mezi sebou navzájem. Za tímto účelem dotazovací jazyk umožňuje použití logických operátorů. Ty mohou výběrová kritéria sloučit, najít jejich průnik, doplněk nebo negaci. Konstrukce dotazu s kombinací výběrových kritérií předpokládá znalost základních množinových operací, neboť v principu provádí logické operace s dílčími výsledky omezujících podmínek. V praxi potom kombinace více kritérií probíhá použitím dotazovacích konstruktů. To mohou být operátory výrokové logiky, jako A nebo NEBO. Negační operátor pro modifikaci omezující podmínky může být například výraz NE. Při formulaci dotazu je typicky umožněno použití i speciálních symbolů, které jsou interpretovány v konkrétním kontextu výběrové podmínky. Například to může být symbol, jenž při zadání výběrového kritéria pro textové pole nahrazuje libovolný počet znaků abecedy. Tímto způsobem lze specifikovat komplexní podmínky pro vyhledávání v textových polích. Výše uvedené postupy pro specifikaci výběrových kritérií lze mezi sebou kombinovat, a proto databázový modul kompetenčního modelu ECDL požaduje pracovní postupy pro jejich specifikaci jak v textovém režimu, tak v režimu grafického zobrazení, jejich přidávání, úpravu a odstranění. Kromě operace selekce lze na tabulky aplikovat projekci. Ta umožňuje výběr pouze určitých sloupců z tabulky.

Při návrhu aplikační vrstvy databázového systému se můžeme setkat se způsobem zobrazení údajů v definovaném rozložení, kdy je najednou zobrazen pouze jeden záznam. Tento způsob zobrazení záznamu nazýváme formulář a jedná se o základní nástroj pro korekci detailní práce se zobrazeným záznamem. Tento druh zobrazení je specifický tím, že můžeme věnovat dostatek prostoru zobrazovacího zařízení jednotlivým polím záznamu.

Kromě toho je možné navrhnout specifický prvek formuláře tak, abychom mohli navrhnout konkrétní komponentu grafického rozhraní určeného pro úzce profilovaný účel. Příkladem implementace výše uvedeného principu do aplikační vrstvy databázového systému tak může být ovládací prvek formuláře, který při aktivaci zobrazí kalendář a dovolí výběr data. V rámci databázového modulu kompetenčního modelu ECDL je primární pozornost soustředěna na návrh struktury formuláře, specifikace formátu a vlastností jednotlivých zobrazených polí a schopnost implementovat navigační prvky. Pod navigačními prvky formuláře je zde myšlena komponenta, která dovoluje pohyb po záznamech tabulky. Základní prvky, jež mohou být implementovány v navigačním modulu, je přesun na první, poslední, předchozí nebo následující záznam tabulky. Dále to může být funkcionalita umožňující vložení nového záznamu do tabulky nebo vymazání existujícího záznamu.

Při specifikaci rozvržení formulářů a jeho grafické podoby je zároveň možnost použít kromě definice jednotlivých polí i jednotnou úpravu záhlaví a zápatí formuláře. Tím lze vytvořit pravidlo vzhledu, které bude shodné pro všechny zobrazované záznamy.

Při práce s databází se kromě návrhu vnitřního uspořádání, aplikační vrstvy nebo znalosti vnitřních principů fungování setkáváme i s oblastí věnovanou prezentaci informací. Ta je zaměřena na pracovní postupy spojené jednak se stanovením kritérií pro výběr údajů z databáze, ale i na možnosti specifikace formy informací, jež mají být prezentovány. Specifickou strukturou, která slouží k prezentaci vybraných údajů z datové báze, může být tzv. sestava. Její primární určení je přehledně zobrazovat údaje z databáze, umožnit export dat z databáze nebo poskytnout rozvržení pro tisk dat. Sestava se obdobně jako formulář skládá z definice struktury zobrazovaných údajů, dále pravidel pro formát zobrazovaných údajů, ale může obsahovat i omezující podmínky výběru údajů anebo shodné grafické prvky pro každou stránku, jako je záhlaví a zápatí.

U každého výše uvedeného prvku existuje množina podrobnějších vlastností, které lze specifikovat. Domnívám se, že nejčastější využití databázové sestavy je v oblasti analytického pohledu na obsažená data anebo přehledové sestavy sloužící primárně k tiskové prezentaci. Důležitou specifickou vlastností je především však možnost vkládání grafických prvků, jako jsou grafy nebo diagramy. Hlavní princip vytvoření sestavy je uspořádané zobrazení údajů

z databáze podle specifikované šablony. Dovednosti, které souvisejí s návrhem šablony sestavy, jsou především schopnost specifikace výběrových kritérií, kritérií pro uspořádání záznamů, kritérií pro seskupování záznamů a definice agregačních pravidel. Kromě toho je v rámci specifikace rozložení šablon důležitá schopnost uspořádat datová pole a definovat záhlaví anebo zápatí sestav.

V oblasti agregačních pravidel jsou důležité dovednosti spojené se schopností vytvářet podmínky pro seskupování údajů, tvorbu součtů, minim, maxim, průměrů a počtů výskytů. Za účelem možnosti použití výsledné sestavy v jiném aplikačním prostředí programu je nutné umět sestavu exportovat nejenom v její grafické podobě, ale i ve formátech určených pro výměnu dat, jako je textový formát, formát XML, formát HTML nebo formát jiné datové struktury.

Jako poslední oblast dovedností je v rámci databázového modulu ECDL definována oblast tisku informací o uspořádání a obsahu databázových objektů. Pro každý jednotlivý objekt databázové aplikace je možné definovat pravidla pro jeho tisk. Základní vlastností v této oblasti je změna orientace tisku (tisk na výšku, tisk na šířku) a formát papíru. Změna rozložení tiskového objektu musí být provedena tak, aby korespondovala se strukturou objektu, a došlo k optimálnímu rozvržení na stránce. Kromě toho je nutné zvládnout specifikaci výběrových podmínek a dovednost předpokládat objem tisku s ohledem na stanovení pravidel pro tisk pouze vybraných záznamů nebo záznamů všech.

Výše uvedený výčet kompetencí a kompetenčních oblastí tvoří standard kompetenčního modelu ECDL. Aby bylo možné zvládnout všechny výše uvedené oblasti kompetencí databázového modulu ECDL, je podle mého názoru nezbytné zvládnout nejenom terminologii a teoretické základy databázového myšlení, ale kromě toho je zapotřebí mít potřebnou praxi s konkrétním návrhovým prostředím databázových systémů. Koncepce celého modulu v sobě zahrnuje nejenom uživatelský přístup k používání aplikační vrstvy databázového prostředí, ale i pohled na konceptuální a datovou vrstvu. Kromě testování uživatelských kompetencí tak ověřuje i základní kompetence databázového specialisty. Cílem modulu je ověřit komplexní znalost práce s databází od vytvoření přes modifikaci až po analýzu struktury databáze. Zvládnutí všech výše uvedených dovedností musí kandidát o udělení certifikace v příslušném modulu dokázat znalostí teoretických konceptů a splněním zadaných úkolů v praktické zkoušce.

3.6 Standard Qulification and Curriruculum Authority

ECDL je standard Evropské unie, o jeho složení se může pokusit kdokoliv. Odlišnou kompetenční základnou tvoří Britský standard kurikula pro oblast ICT. Zásadním rozdílem činí právě zaměření standardu. Databázový modul kompetenčního modelu ECDL je směřován na evaluaci kompetencí v oblasti ICT. Oproti tomu kompetenční model QCA je koncipován jako podpůrný prostředek propedeutické úrovně ICT na základní škole. Standard vymezuje vzdělávací cíle, obsah výuky a očekávanou znalostní hladinu žáků. Je součástí Britského národního kurikula a jeho hlavním cílem je „být vodítkem pro správné rozhodování učitelů, být příkladem toho, jak může vypadat praktická aplikace vzdělávacího standardu“ (QCA, 2003).

Hlavní pozornost v rámci disertační práce směřuji do 1. a 2. klíčové fáze. Ty jsou dále členěny do úrovní. Jednotlivé kompetenční úrovně v rámci prvních dvou klíčových fázích odpovídají prvnímu a druhému stupni základní školy. Konkrétně je ve standardu uvedeno, že 2. úroveň odpovídá druhé třídě základní školy, 3. úroveň čtvrté třídě základní školy a 4. úroveň šesté třídě základní školy. Pro jednotlivé klíčové fáze jsou určeny hlavní cíle. Pro první klíčovou fázi je to dosáhnout u žáků rozvoje schopností pro další průzkum ICT, rozvoj kreativního myšlení a základní pochopení hardwaru a softwaru. Ve druhé klíčové fázi jsou hlavními cíli naučit žáky používat ICT nástroje, položit základy vědeckého myšlení a zabývat se otázkou kvality informací. Z kompetenčního pohledu je model rozvoje ICT kompetencí rozdělen do tří oddělených oblastí, ve kterých jsou definovány konkrétní postupy, jež vedou k jejich rozvoji a zdokonalování. Databázové myšlení je v případě standardu QCA založen na osvojení dovedností v oblastech sběru údajů, návrhu datové struktury a prezentaci informací. Práce s databází je ve výše uvedeném pojetí informační výchovy pouze dílčí oblastí standardu. Tvoří ale nedílnou součást souboru komplexních dovedností, jež vedou k rozvoji práce s informačními technologiemi. Kurikulum tak reaguje na požadavky informační společnosti rozvojem informační gramotnosti na úrovni primárního setkání s prostředky informačních a komunikačních technologií. Metody směřující k rozvoji informační gramotnosti jsou založeny na konstrukci kompetenčních základů dílčích dovedností, jež jsou základem pro zvládnutí práce s databázemi. V oblasti kognitivních struktur žáka je stěžejním zájmem rozvoj kompetence pro manipulaci se strukturou, jež se stává prostředníkem pro uchopení více než jedné informace. Kurikulum tak implementuje koncepci pro zahájení přípravy rozvoje databázového myšlení již na prvním stupni základní školy. Rozvoj

databázového myšlení rozvíjí mozaiku hierarchicky uspořádaných kompetencí nezbytných pro zvládnutí práce s databází.

Analýza cílů výuky standardu QCA rozkryla důležitost návaznosti jednotlivých kompetencí. Jednotlivé kompetence jsou vzájemně hierarchicky uspořádány. Na základě hierarchie lze stanovit předpoklady pro schopnost pracovat v konkrétním databázovém prostředí. Kromě toho lze z hierarchie jasně určit kognitivní náročnost dílčích kompetencí a zároveň z nich vyplývá optimální pořadí jejich osvojování. Navíc je ve standardu databázové myšlení integrováno do kontextu rozvoje informační gramotnosti jako celku. Rozvoj databázového myšlení tak umožňuje i rozvoj ostatních kompetencí informační gramotnosti. Standard je koncipován tak, aby umožnil i rozvoj mezioborových vazeb.

Při analýze systému učiva databázového myšlení jsem se zabýval jeho vhodným rozdělením na dílčí podsystémy. Při jejich určování jsem zvolil klíčovou kompetenční oblast. Tou je rozvoj databázového jazyka jako prostředku pro komunikaci s databází. Další důležité podsystémy se nalézají v oblasti konstrukce struktury datové báze a prezentace údajů z datové báze. Konkrétní rozdělení kompetencí databázového myšlení odpovídá standardu QCA. Kategorizace kompetencí se orientuje na směry rozvoje dovedností vztažených k oblastem komunikačního prostředku (dotazovacího jazyka) pro manipulaci s datovou bází, směru rozvoje kompetencí pro konstrukci struktury datové báze a směru rozvoje kompetencí pro prezentaci údajů z datové báze. Zvládnutí komunikačního prostředku je založeno na schopnosti porozumět principům fungování konceptuální vrstvy databázového systému a dovednosti vyjádřit neprocedurální konstrukt pro manipulaci s ním. Kompetenční rovina schopnosti strukturovat datovou bázi zahrnuje především znalost fyzické vrstvy databázového systému a dovednost správně převést požadavky na vnitřní uspořádání datové báze do jeho definice. Kompetence pro prezentaci informací z datové báze jsou založeny na vhledu do podstaty strukturace datové báze na aplikační úrovni a schopnost vytvořit neprocedurální konstrukt pro získání relevantních údajů.

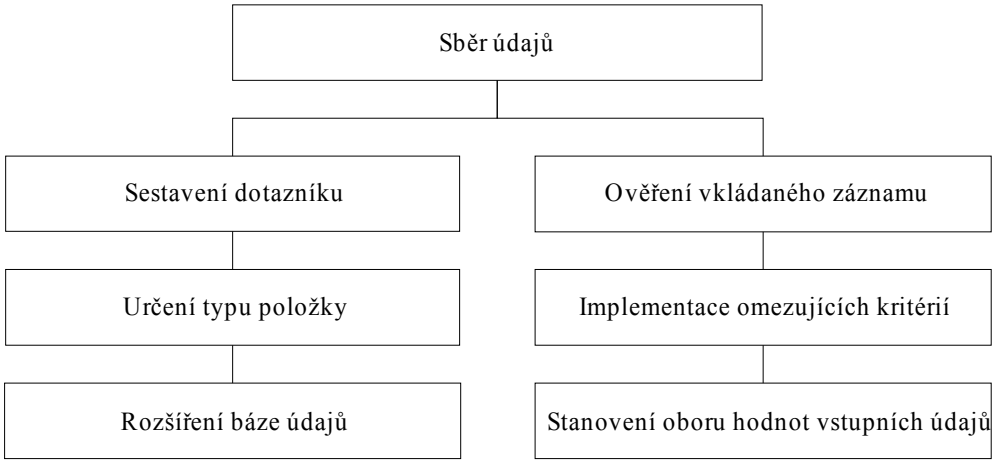
Cílem této kapitoly je rozpracovat konečné uspořádání kompetencí databázového myšlení v souladu s výše uvedenou kategorizací. Dílčím cílem kapitoly je nastínit hierarchickou strukturu cílových kompetencí tak, aby bylo možné uvažovat obecněji o jejich uspořádání bez vztahu ke konkrétní implementaci databázového prostředí. V souvislosti s konstrukcí hierarchické struktury cílových kompetencí jsem provedl analýzu výukových cílů rozborem požadované výsledné struktury hierarchie kompetencí žáka v dané oblasti a její visuální reprezentaci.

První kompetenční oblast je sestavena z dílčích kompetencí pro zajištění sběru údajů. Je charakterizována předpokladem znalosti významu pojmů dotazník, datový typ, obor vstupních hodnot a omezující kritérium. Výukovým cílem je naučit žáka osvojit si princip pracovního postupu pro navržení funkčního a logicky správného dotazníku podle zadání. Dílčím cílem je naučit žáka zvládnout princip návrhu ověřování vkládaných údajů. Dále je kompetence k návrhu dotazníku možné rozdělit na kompetence spojené s návrhem položek dotazníku a kompetence související se schopností určit údaje, které lze do dotazníku vkládat. Aby bylo možné navrhnout princip ověřování položek dotazníku při vkládání záznamu, je nutné umět určit datový typ vkládaných údajů.

Kat.	**Název**	**Popis**
1.	Sběr údajů	Žák dovede sestavit dotazník a zajistit ověřování vkládaných údajů. Jedná se o schopnost složenou ze schopností sestavit dotazník a navrhnout metodu pro ověřování vkládaných údajů.
1.1.	Sestavení dotazníku	Žák dovede navrhnout položky dotazníku a umí určit jejich typ.
1.1.1.	Určení typu položky	Žák umí určit charakter vkládaných údajů.
1.1.1.1.	Rozšíření báze údajů	Žák ví, jak rozšířit bázi údajů o nový záznam.
1.2	Ověřování vkládaného záznamu	Žák si uvědomuje možnost chyb v datech a ví, jak navrhnout implementaci omezujících kritérií.
1.2.1.	Implementace omezujících kritérií	Žák dokáže určit obor hodnot údajů zadávaných do dotazníku. Umí navrhnout podmínku vylučující uložení chybného záznamu.
1.2.1.1	Stanovení oboru hodnot vstupních údajů	Žák dokáže stanovit pro položku dotazníku všechny možné hodnoty, které může nabýt. Schopnost je založena na určení vlastnosti reálného předmětu a určení jejích možných hodnot.

Tabulka 2 Kompetence ke sběru údajů dle QCA

S principem návrhu metody pro ověřování položek je úzce spojená dovednost implementovat dílčí kritéria, jež umožňují ověřit položky. Při implementaci validačních kritérií je nezbytné nejdříve určit obor hodnot vkládaných údajů a na jeho základě sestavit podmínku, jež umožní zamezit vložení nesprávného údaje. Rozvoj kompetence pro stanovení datového typu je založena na znalosti pracovního postupu pro přidávání prvku do množiny. Při něm dochází k myšlenkové operaci určení typu vkládaného prvku a na jeho základě rozhodnutí o korektnosti operace vložení nového prvku do množiny. Právě rozvoj schopnosti pracovat s myšlenkovým aparátem rozhodování o správnosti operace přidávání nového prvku do množiny je v rámci níže uvedené hierarchie kompetencí stanoven jako základní. Na stejné, základní úrovni je i dovednost pro stanovení oboru hodnot vstupních údajů. Ta úzce souvisí s myšlenkovou operací určení vlastností reálného předmětu. Konkrétně se jedná o vymezení hodnot, jež může vlastnost stanoveného předmětu nabýt.



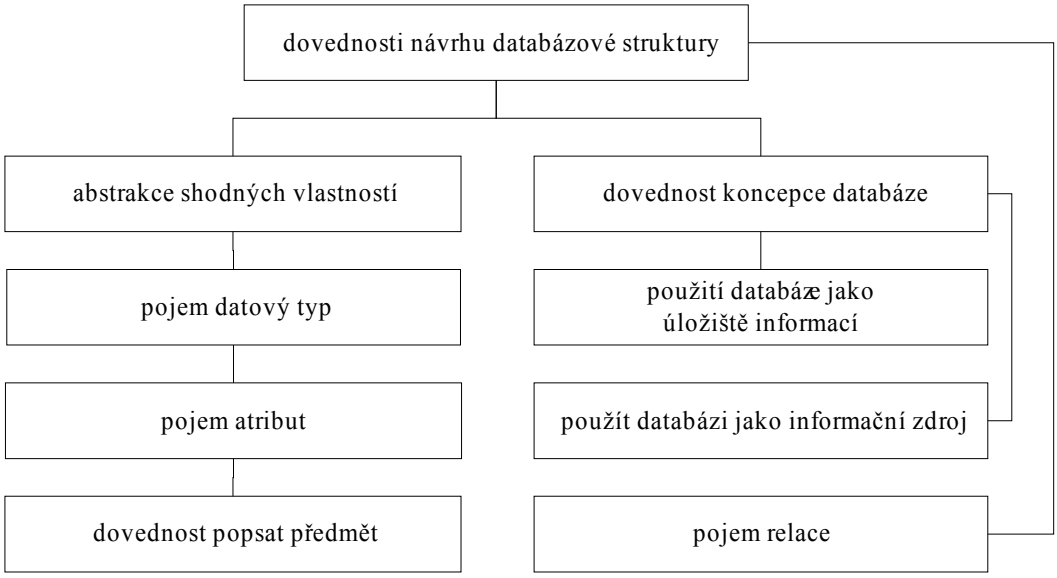
Obrázek 6 Hierarchie kompetencí ke sběru údajů

Jak uvádí Mudrák (Mudrák, 2007), „jádro dovedností navrhovat databázové schéma je položeno ve schopnostech vhodně pojmenovávat abstraktní entity reprezentující jistý výsek reálného světa a plně si uvědomovat kvalitativní znaky vztahů mezi nimi. Struktura pojmenovaných entit a jejich vzájemných vztahů se zobrazuje v různých diagramech a pro práci s databázemi je nezbytná dovednost diagram rozumně a s reprezentovanou strukturou dále manipulovat“.

Kategorie.	**Název**	**Popis**
2.	2. Dovednost návrhu databázové struktury	Žák je schopen navrhnout entity a jejich relace podle koncepce databáze.
2.1.	2.1. Abstrakce shodných vlastností	Žák dokáže abstrahovat vlastnosti předmětů do shodných skupin a na jejich základě definovat a pojmenovat skupinu předmětů se shodnými vlastnostmi.
2.1.1.	2.1.1. Pojem datový typ	Žák je schopen určit datový typ atributu entity.
2.1.1.1.	Pojem atribut	Žák dokáže vyjádřit vlastnosti skupiny v podobě atributů entit.
2.1.1.1.1.	Dovednost popsat předmět	Žák umí popsat předmět s použitím výrazů vybraných z relevantní slovní zásoby.
2.2.	Pojem relace	Žák ví, jak určit vztah mezi entitami.
2.3.	Dovednost koncepce databáze	Žák dokáže přizpůsobit strukturu databáze nejenom požadavkům vyplývajícím z úlohy databáze jako úložiště informací, ale i nárokům stanoveným na základě role databáze jako informačního zdroje.
2.3.1.	Přizpůsobit databázi jako úložiště informací	Žák umí přizpůsobit databázovou strukturu v souladu se strukturou dotazníků určených pro sběr dat.
2.3.2.	Přizpůsobit databázi jako informační zdroj	Žák ví, jak uspořádat strukturu databáze v souladu s úlohou databáze ve smyslu informačního zdroje sloužícího především jako podklad pro prezentaci informací.

Tabulka 3 Kompetence pro návrh databázové struktury

Mudrák (Mudrák, 2007) dále uvádí, že: „dovednosti pro modelování systémů schopnosti se opírají o analytické a systémové myšlení, jako je způsobilost uvědomovat si daný systém jako celek i jako seskupení jeho částí, schopnost vhodně a výstižně pojmenovávat identifikované komponenty systému a dovednost zaznamenávat jejich vztahy v dohodnuté formální symbolické notaci. Zároveň se očekává vhled do modelované problematiky a její grafické znázornění zvolenou notací.“ Schopnosti pro rozvoj databázového myšlení jsou tak kromě oblasti dotazování, návrhu a prezentace úzce spojeny s klíčovou kompetencí manipulovat s kognitivní reprezentací struktury a manipulaci s ní. Výše uvedené pojetí je zaměřeno do oblasti praktické aplikace konceptuálního modelování databázové struktury, lze však u něj určit úzkou souvislost s rozvojem dovedností pro transformaci reálných předmětů a skupin předmětů do vyjádření v podobě entit. Rozvoj transformační oblasti je základním prvkem pro rozvoj návrhu databázové struktury. Podrobnější rozbor konstrukce hierarchického uspořádání kompetencí pro návrh datové struktury je graficky zobrazen na diagramu kompetenční hierarchie návrhu databázové struktury.



Obrázek 7 Hierarchie dovednosti návrhu databázové struktury

Jak již bylo uvedeno, kompetence pro prezentaci informací z datové báze jsou založeny na schopnosti vhledu do podstaty strukturace datové báze na aplikační úrovni. To v sobě obsahuje především schopnost překladu požadavku na získání konkrétních informací z datové báze do smysluplného a správného neprocedurálního konstruktu. Kromě toho je však podstatnou součástí vhledu do strukturace schopnost zpracovat výstup, jenž je výsledkem dotazu, který žák položí. Formulace neprocedurálního konstruktu je v kompetenční oblasti

prezentace informací chápána jako schopnost formulovat dotaz v dotazovacím jazyce. K tomu, aby žák mohl být schopen korektně formulovat dotaz, je nezbytné, aby byl schopen kognitivních operací projekce, selekce, seskupení a uspořádání. Ty jsou základem pro predikci výsledku dotazu. Vzhledem k vyjadřovacím schopnostem dotazovacího jazyka je tedy nutné porozumět relaci mezi strukturou záznamu a výše uvedenými operacemi. Žák musí vědět, že při použití projekce je ze záznamu v databázi vybrán pouze určitý atribut. Současně musí mít žák znalost analogie použití projekce jako stanovení kritéria pro výběr určitých vlastností ze všech vlastností reálného předmětu. Selekce je určena ke stanovení výběrového kritéria a slouží k omezení výběru záznamů na jejich podmnožinu, jež zadané omezující podmínce vyhovuje.

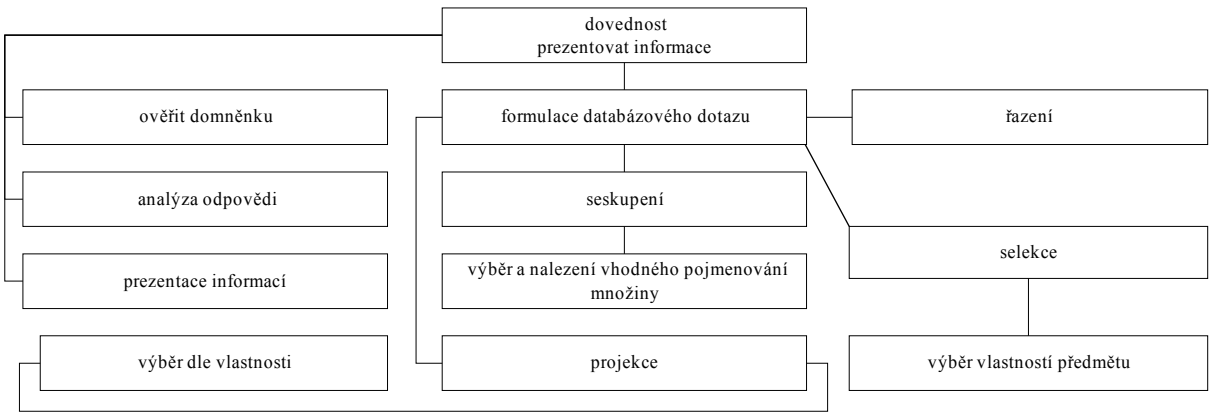
Kromě toho musí být žák schopen uvažování pomocí analogie selekce jako výběru omezené podskupiny předmětů ze všech předmětů obsažených v množině. Další kognitivní operací nezbytně potřebnou k dosažení rozvoje schopnosti správné formulace databázového dotazu je znalost principu seskupování. Jedná se konstrukt, který umožňuje podle zadaného kritéria seskupit záznamy datové báze tím, že pro určitý atribut entity specifikuje sdružovací pravidlo. Na jeho základě se provede seskupení položek, jež vyhovují stanovené podmínce. Navíc je nezbytné, aby žák věděl, že pokud provede seskupení záznamů, je nutné definovat agregační funkci, která provede nad skupinou záznamů funkci. V návaznosti na předměty reálného prostředí musí mít žák osvojený koncept sdružování předmětů na základě stanovených vlastností. Může tak vytvořit ze skupiny předmětů několik nových skupin tím, že pro novou skupinu zvolí shodnou vlastnost předmětů, již musí splňovat. Další podstatnou kognitivní operací pro formulaci dotazu pro práci se záznamy v datové bázi je specifikace kritéria pro řazení předmětů, jež mají být výstupem dotazu. Jedná se o specifikaci pravidla pro uspořádání výstupních prvků. Žák musí znát analogii s uspořádáním reálných předmětů.¨

Kategorie	**Jméno**	**Popis**
3.	Dovednost prezentovat informace	Žák dokáže ověřit svou domněnku sestavením dotazu v dotazovacím jazyce. Získané informace umí nejenom prezentovat, ale i analyzovat. Zároveň zná význam údajů a dokáže vysvětlit jejich význam.
3.1.	Formulace databázového dotazu	Žák umí formulovat dotaz v dotazovacím jazyce na základě kombinace projekce, selekce, seskupení a uspořádání.
3.1.1.	Projekce	Žák vybere pouze některé vlastnosti záznamu.
3.1.1.1.	Výběr vlastnosti předmětu	Žák dovede vybrat pouze určitou vlastnost předmětu na základě stanovených kritérií.

Kategorie	Jméno	Popis
3.1.2.	Selekce	Žák vybere pouze některé záznamy v databázi.
3.1.2.1.	Výběr dle vlastnosti	Žák vybere ze všech předmětů pouze ty, jejichž vlastnost vyhovuje stanoveným podmínkám.
3.1.3.	Seskupení	Žák určí vlastnost předmětu pro seskupení položek v databázi a skupinu pojmenuje.
3.1.3.1.	Výběr a nalezení vhodného pojmenování množiny	Žák dovede vybrat předměty, které vyhovují stanoveným kritériím, a umístit je do množiny předmětů se shodnou vlastností.
3.1.4.	Řazení	Žák stanoví pravidlo pro uspořádání položek v databázi na základě vybraného předmětu postupně od nejmenšího k největšímu a naopak.
3.2.	Ověřit domněnku	Žák dokáže analýzou odpovědi databáze ověřit svou domněnku a své zjištění vhodnou formou prezentovat.
3.2.1.	Prezentace informací	Žák dokáže navrhnout formu prezentace s ohledem na publikum, pro které je určena.
3.2.2.	Analýza odpovědi	Žák dokáže analyzovat odpověď databáze.

Tabulka 4 Kompetence oblasti dovednost prezentovat informace

Výše uvedené kognitivní operace jsou předpokladem pro správnou formulaci otázky, jež by mohla potvrdit nebo vyvrátit žákovu domněnku. Cílovým stavem žákových kompetencí je nejdříve přeložit požadavek na ověření domněnky do dotazovacího jazyka. Následně provést zadání dotazu a uchopit odpověď datové báze. Nakonec stojí před žákem úkol odpověď převzít, analyzovat relevanci odpovědi vzhledem k formulované domněnce a získanou informaci ve vhodné formě prezentovat s ohledem na potřebu cílové skupiny, jíž je výsledek určen.



Obrázek 8 Kompetenční složení dovednosti prezentovat informace

V další kapitole se budu blíže zabývat standardem ECDL, který sice nesouvisí s QCA, ale stupněm obtížnosti na něj navazuje. Zatímco QCA je úrovní na úrovni základní školy, obtížnost ECDL lze zařadit na úroveň školy střední.

3.7 Rozvoj databázového myšlení ve standardech ICT

Kompetenční modely ECDL a QCA jsou svým zaměřením velice rozdílné. Zatímco koncept ECDL je standardem pro evaluaci databázových schopností, je kompetenční model QCA určen jako nástroj pro podporu výuky učitelů pro žáky, u nichž dochází k primárnímu rozvoji databázového myšlení. Rozdílné znaky lze sledovat i v oblasti zaměření na určitý produkt. Zatímco ECDL je velice konkrétně specifikován a stěžejním principem je stanovení evaluačních kritérií osvojení pracovních postupů, je QCA svým zaměřením naopak specifikováno obecně a hlavní náplní je rozvoj konceptů bez ohledu na konkrétní produkt. V oblasti strukturálního rozdílu mezi standardem QCA a ECDL lze však najít shodné znaky. Těmi je především rozdělení konceptuálních celků do oblasti ukládání dat, změny struktury dat a prezentace dat. Dle výše uvedených oblastí lze stanovit klíčové kompetence v dané oblasti se zaměřením na syntézu dílčího cíle daného standardu. Databázové myšlení lze dále členit na oblasti zvládnutí databázové terminologie, konceptuálního modelování a dotazovacího jazyka.

V rámci britského kurikula vláda vydala Národní strategii, která soustředí jednotlivé učební celky do tzv. klíčových stupňů (QCA, 2007). Kritéria splnění dílčích stupňů jsou založeny na procentuálním množství studentů, kteří jsou daný stupeň schopni zvládnout. Učební rámec publikovaný organizací QCA je založený na pilotním programu a je výsledkem osvědčených postupů. Koncepce ale vznikala již dříve, kdy se klíčová úroveň 3 rozvíjela od září 2001, kdy byly uvedeny dva standardy pro angličtinu a matematiku. Tři další standardy, zaměřené na vědu, ICT a základní předměty byly uvedeny v 2002-03.

Dílčí cíle je možné členit podle stupně zvládnutí databázového myšlení. Jednotlivé úrovně jsou uspořádány do hierarchie a vyšší úrovně využívají kognitivní procesy nižších úrovní. Nejnižší úroveň je charakterizována hledáním možností a formy sdělení. Zároveň do ní patří rozhodování o výběru možnosti umožňující nejpřesnější sdělení významu. Pro druhou úroveň je typické hledání informací a cílené využití znalostí k dosažení konkrétních výsledků. Třetí úroveň je charakteristická rozvojem myšlenek a řešením problémů. Čtvrtá úroveň je zaměřená na organizaci, reorganizaci a analýzu informace. Pátá úroveň je charakteristická rozvojem osvojených dovedností především v oblasti rozvoje schopnosti formulace složité otázky

a rozvojem dovednosti pro hledání informace.

Obecnou studií rozvoje programovacího jazyka se zabývá Shivers (Shivers, 2008). Podle jeho názoru by studium počítačových jazyků mělo být zaměřeno na skutečné porozumění základních principů fungování počítačů. Zdůrazňuje fakt, že počítačový jazyk je primárně určen pro lidi, a to především jako nástroj pro popis výpočtů. Počítač je však schopen zpracovat pouze strojový kód. Forma programovacího jazyka zachovává jeho určení jako nástroj člověka pro rozhodování o návrhu a struktuře výpočtů. Hlavní mechanismy, které to umožňují, jsou modularita, abstrakce, rozsah, rekurze atp. Ty jsou určeny lidem pro překonání kognitivních nedostatků. Zároveň slouží ke zvýšení kognitivní síly člověka tak, aby byl schopen zkonstruovat program, jehož hranice nejsou typicky dané pouze rychlostí procesoru nebo diskové kapacity, ale právě omezenou schopností člověka zvládnout komplexnost.

V rámci analyzovaného standardu QCA lze pět vymezit pět úrovní rozvoje databázového myšlení. Jsou uspořádány vzestupně podle kognitivní náročnosti. Úrovně jsou pojmenovány dle jejich hlavního cíle. Jmenují se propedeutická, nácvičná, expansivní, stabilizační a integrační úroveň. Propedeutická úroveň je zacílena na zajištění porozumění jednoduchým ICT nástrojům. Nácvičná úroveň se vyznačuje cíleným používáním ICT k dosažení konkrétních výsledků. Expansivní úroveň je typicky charakterizována použitím ICT pro rozvoj myšlenek. Stabilizační úroveň je zaměřena na evaluaci informačních zdrojů. Integrační úroveň je zaměřena na kombinaci ICT nástrojů v rámci komplexního řešení.

Do jednotlivých kategorií jsem umístil dovednosti v pořadí od nejjednodušších k nejsložitějším. Tedy v pořadí, ve kterém by měly být osvojovány. Cílem jejich zvládnutí je rozvoj databázového myšlení, dle QCA (QCA, 2003):

1) Propedeutická úroveň:
    a. použití slovní zásoby,
    b. informace okolo nás,
    c. charakteristika a třídění.
2) Nácvičná úroveň:
    a. hledání informací,
    b. otázky a odpovědi.
3) Expansivní úroveň:
    a. úvod do databází.
4) Stabilizační úroveň:
    a. větvené databáze.
5) Integrační úroveň:
    a. analýza údajů.

Výše uvedené stupně rozvoje databázového myšlení dovolují rozvíjet schopnost a dovednosti pro organizaci, vyhledávání, řazení a zobrazování informací. To umožňuje zkoumat jejich vztahy, hledat zákonitosti a ověřovat formulované hypotézy o informačním uspořádání a struktuře. Jedním z východisek a teoretický základ, o který se rozvoj databázového myšlení opírá, jsou principy výuky programovacích jazyků. Rozdíl mezi výukou principů procedurálního programování a rozvojem databázového myšlení je především v přístupu řešení problému. Zatímco procedurální programování se zabývá formulací způsobu, jak problém vyřešit, tak databázové myšlení je založen na hledání na otázky, co má být výsledkem řešení.

Při hledání cíle rozvoje databázového myšlení učitelů ICT je možné se opřít o teoretické základy uvedené v pracích věnovaných teoriím výuky programování. O rozvoji cílů a směru rozvoje nástrojů pro výuku programování podává zprávu již Milner (Milner & Wildberger, 1977) a Camstra (Camstra, 1977), kteří se zabývali zlepšováním instrukčních nástrojů. V souvislosti s vlivem nástupu mikropočítačů se Hazari (Hazari, 1991) pokusil definovat funkční model cílů výuky informační gramotnosti. Cílem vyučování a strukturou znalosti v paměti se zabýval i Gagne (Gagne & White, 1978). Z jeho závěrů lze usuzovat, že rozvoj

databázového myšlení je spojen i s dalšími komplexními dovednostmi a znalostmi. Mezi ty se řadí například matematické schopnosti, především v oblasti učení logických pravidel a jazykových schopností. Domnívá se, že jazykové kompetence jsou spojené především se schopností učit se syntaktická pravidla. Bottino (Bottino, 1992) se věnoval pedagogickému hledisku různých přístupů k programování. Zkoumal rozdíl přínosu učení dvou různých programovacích jazyků. Thornburg (Thornburg, 1988) se domníval, že základní rozdíl mezi vysokoúrovňovými programovacími jazyky nespočívá v různé syntaxi, gramatice nebo slovnících, ale v jejich odlišných metaforách. Pokud tedy vyučujeme databázové myšlení, potažmo dotazovací jazyk, jedná se pouze o odlišnou metaforu, pohled na datové struktury.

V minulosti se v rámci vytyčování cílů potýkali vědci s obecnějšími problémy. Například Kurtz (Kurtz & Adams, 1988) viděl smysl rozvoje databázového myšlení jako způsob zdokonalení kompetencí k řešení problému. Je to jeden z pohledů, ve kterém je strukturní myšlení považováno za prostředek k rozvoji dalších kompetencí. Problematiku obecných kompetencí rozpracoval do podoby principu expanze (expanduje myšlenku z její jednoduché podoby do složitějších podob) a integrace úrovně spojující několik konceptů na stejné úrovni obtížnosti. Výše uvedené koncepty ukazuje na daných příkladech: kontrola toku, rekurze a skrývání informací. Výsledkem svého výzkumu přispěl k nalezení rovnováhy mezi schopnostmi řešit problém a osvojováním schopností práce s počítačem. Sauter (Sauter, 1986) předpověděl nezbytné znalosti pro rozvoj myšlení v dotazovacím jazyce. V době psaní svého článku sledoval posun náhledu na učení počítačového jazyka. Z jeho závěrů lze usuzovat na spojení matematických schopností se schopností učit se logická pravidla a jazykové znalosti se znalostí syntaktických pravidel.

Jedny z častých otázek ve výzkumu výuky programování se týkají potřebných znalostí pro studenty učitelství. Této problematice se věnoval i Kushan (Kushan, 1994). V oblasti přípravy budoucích učitelů programování sestavil přehled pokusů definic kvality instruktorů programování v rámci různých expertních skupin, které se pokusily ustanovit, jaké znalosti by měli mít budoucí učitelé programování. Profesionální vědci sestavili ukázkové studijní programy. Několik států v USA vyvinulo standardy pro certifikáty ve výuce počítačových kurzů a univerzity navrhly programy studia pro přípravu učitelů informatiky. Tyto programy se však od sebe navzájem velice lišily. Již Kushan (Kushan, 1994) deklaruje trojici kompetenčních kategorií nároků na ICT učitele. Je to kategorie rozvoje kompetencí k řešení problémů, pedagogických znalostí a programátorských znalostí. Mezi koncepty, které považuje za stěžejní, je znalost syntaxe programovacího jazyka, sémantická znalost, přehled o strukturovaném programování, top-down návrhu, modularitě, tvorbě dokumentace, znalost

návrhových nástrojů, techniky odlaďování a simulace testování. Zároveň sestavil historický přehled strukturálních změn znalosti učitelů v letech 1980–1995 a rozdělil je do tří fází vývoje. V první fázi se na středních školách vyučovala především syntaxe. Hlavním důvodem byl fakt, že většina učitelů byli programátoři, a proto vyučovali především psaní programů. Důležité pro ně bylo, aby program fungoval a produkoval správný výstup. Počátek druhé fáze vymezil publikací Papertovy knihy Mindstorms (Papert, 1980). Ta byla převratná především v názoru, že „díky tomu, že žáci budou učit počítače myslet, mohou rozšířit své znalosti o tom, jak ony samy přemýšlí“. Na základě této teorie vzniklo celosvětově mnoho různých pokusů o výuku pomocí instrukcionálních metod vedoucích k řízenému objevování v nestrukturalizovaném prostředí. Výzkum v polovině osmdesátých let ale poukázal na fakt, že takové učení bylo pro žáky příliš složité, a přenos učení do jiných oblastí se zdál minimální.

Třetí fáze byla charakterizována návratem k přímějším instrukcionálním metodám vedeným učitelem. Kushan se pokusil formulovat výsledky výuky programování (Kushan, 1994). Zdůraznil domněnku, že hlavní přínos učení programování spočívá ve zlepšení schopnosti řešit problém, rozvíjí poznatky o vlastním myšlení, procvičuje znalosti z matematiky, rozšiřuje poznatky o sociální interakci a napomáhá uvědomění hodnot. Tím potvrzuje domněnku, že rozvoj databázového myšlení může být nejenom prostředkem pro rozvoj technické dovednosti. Ostatně tuto domněnku potvrdil již Soloway (Soloway, 1986) výrokem: „Programování rozvíjí schopnost řešit problém, protože podporuje nutnost nadhledu z procedurálního hlediska.“ Podle Callistera (Callister & Burbules, 1990) se musí kurikulum studentů učitelství ICT revidovat z pohledu budoucího uplatnění nabytých znalostí. Navrhuje přesunout pozornost od výuky obecných kompetencí ke konkrétním aplikačním doménám. V rámci výuky obecných předmětů navrhuje zavést počítače. Zároveň doporučuje u studentů učitelství ICT brát v potaz používání počítačů i sociologické a psychologické hledisko jejich vlivu na učení. Otázkou přípravy na používání počítačů se zabývá Handler (Handler, 1993), který si položil otázku, jakým způsobem je nutné definovat připravenost učitelů.

Fisher (Fisher, 2008) se zabývá otázkou, proč potřebujeme více než pouze jeden programovací jazyk. Myslí si, že schopnost rozhodovat se by měla být předána zároveň s informací o tom, jaký programovací jazyk použít v dané situaci. Dokazuje, že každý programovací jazyk je určený k řešení specifického druhu problémů a je založen na různém teoretickém základě. Pro dotazovací jazyk vidí původ jeho charakteru v relační algebře. Z toho vychází její názor, že v rámci kurikula by měl být vyučován více než jeden programovací jazyk jako základ kompetence pro učení nových programovacích jazyků, a to

díky jevu, že každý nový programovací jazyk je snadnější se naučit, pokud již student nějaký programovací jazyk ovládá. Na základě znalosti více než jednoho programovacího jazyka si kromě jiného student osvojuje schopnost určit silné a slabé stránky těchto jazyků. S touto znalostí může student vyhodnotit problém a na základě této znalosti vybrat vhodný programovací jazyk.

Na základě výše uvedených příkladů činností pro rozvoj databázového myšlení se domnívám, že další oblastí, na kterou je nutné se v rámci rozkrytí celé problematiky zaměřit, je oblast rozvoje kompetencí učitelů ICT: Základní stavební kámen ve vzdělávání učitelů vidím v podobě jejich přípravy na budoucí povolání. Dle mého názoru by výuka budoucích učitelů ICT pro účely rozvoje databázového myšlení měla být zaměřena především na principy databázových systémů tak, jak ve svém článku publikoval Jukic (Jukic & Gray, 2008). Zdůrazňuje fakt, že výuka databázového myšlení pro učitele by měla být zaměřena především na jejich přípravu v oblasti znalosti obecných myšlenkových konceptů. Tuto oblast upřednostňuje před orientací na rozvoj aplikovaných technických dovedností. Zároveň se domnívám, že základním principem rozvoje vzdělávací soustavy je zvyšování statusu a profesionality pedagogických odborníků. Výše uvedený princip je zakotven v legislativě. Legislativním pramenem pro oblasti profesního rozvoje učitelů je zákon č. 561/2004 Sb., o předškolním, základním, středním, vyšším odborném a jiném vzdělávání (školský zákon), ale především Zákon č. 563/2004 Sb., o pedagogických pracovnících.

Ke konkrétní náplni kurikula se vyjadřuje i Murray (Murray & Guimaraes, 2008). Ten je ovšem zastáncem jiného směru rozvoje standardního obsahu kurikula kurzu databázových technologií. Je toho názoru, že správným směrem rozvoje je prohloubení odborných znalostí. Navrhuje zahrnout do databázového kurzu pokročilá témata, jako jsou datové sklady, databázová bezpečnost a ladění výkonu databáze. Obdobný názor na obsah databázového kurzu zastává George (George & Valeva, 2006). Považuje za důležité vyučovat v databázovém kurzu kromě základních principů databázových systémů i otázky bezpečnosti.

Udoh (Udoh, 2006) zastává názor, že databázový kurz by měl být rozdělen na dvě části. První část by se měla týkat základů databázových systémů. Tedy relačního modelu, ER modelování, normalizace, dotazovacího jazyka, relační algebry a počtu, organizace úložiště a optimalizace dotazů. V pokročilém kurzu potom navrhuje vyučovat transakce, kontrolu konkurence, zotavení, data warehouse, bezpečnost, distribuované a objektové databáze, data mining a administraci databázových systémů. Dekeyser (Dekeyser et al., 2007) se pro změnu domnívá, že náplň obsahu databázového kurzu nemá za úkol pouze naučit studenta psát gramaticky správné dotazy, ale především překládat otázky v přirozeném jazyce do výrazu v

dotazovacím jazyce.

Zvládnutí technických témat umožní učiteli zvládnout praktickou výuku. Rozvoj databázového myšlení by měl směřovat především k aplikaci nabytých poznatků do výuky. Zároveň by definice kompetenčního rozsahu databázového myšlení měla brát ohled na fakt, že se jedná pouze o dílčí oblasti ICT.

Z toho vyplývá, že databáze musí být vyučovány účinně, efektivně a v omezeném časovém úseku. Z důvodu nutnosti naplnit oba výše uvedené požadavky a vzhledem k faktu, jak obsáhlé je téma databázových systémů, databázového myšlení a souvisejících oblastí, docházím k názoru, že je nutné databázové myšlení rozvíjet, jak nejdříve to jde, a to především v rovině propedeutické. Pokud se tak neděje, lze se domnívat, že pozdní rozvoj kompetencí databázového myšlení může ve svém důsledku vést až k neschopnosti databázové myšlení zvládnout.

Cook (Cook, 2008) pojednává o obtížích spojených se sestavením efektivního kurikula. Uvádí, že jedním z problémů, které je nutné v rámci výuky programování řešit, je indoktrinace. Je to vliv nástrojů, které používáme, na naše myšlení o problémech. Vzniká tak, že student se pokusí nesprávně použít přístup z již naučeného programovacího jazyka na nově naučený programovací jazyk, který je ovšem založený na jiném paradigmatu. Další problém, který identifikoval, je výsledek vlivu kvantity na efekt měřítka. Jedná se o to, že dokud studenti nepoužívají softwarové rozsáhlé a objemné informační systémy, neprojeví se například nedostatky algoritmu, které by se projevily u velkého systému. V neposlední řadě uvádí dilema o směru, kterým by se výuka programování měla ubírat. Proti sobě tady působí dva faktory. První je hloubka výuky a druhý je rozsáhlost.

Holden se ve svém článku (Holden, 2008) zabývá otázkou přiřazení cílů kurzu jednotlivým hodnotícím kritériím pomocí stanovení výsledků učení v rámci kurzu. Postup, který popisuje, by mohl posloužit pro účely hledání vhodného poměru mezi teoretickou průpravou budoucích učitelů ICT a rozvojem jejich praktických dovedností pro práci s databázemi.

Zastávám názor, že kompetence pro zvládnutí databázového myšlení by měly být standardní součástí kompetencí ICT učitelů. Profesní standard učitele, jak jej deklaruje Rýdl (Rýdl et al., 2009), chápe profesní standard kvality učitele jako rámec žádoucích kompetencí a činností v navrhovaných ukazatelích. Jedna z možných cílových pokročilých aplikací kompetencí učitele může být, jak uvádí Seyed-Abbassi (Seyed-Abbassi, 1993) použití projektu ve výuce informačních systémů. Dospěl k názoru, že projekt jako takový rozšiřuje studentovo porozumění teoretickým konceptům, rozšiřuje jeho povědomí o náročnosti a složitosti použití počítačových technologií v praxi. Projektový přístup by mohl být aplikován i v rozvoji

databázového myšlení u budoucích učitelů ICT. To by mohlo dopomoci přiblížit přípravu učitelů reálným nárokům praxe. Příprava by se tak měla odehrávat ve dvou hlavních rovinách. A to v rovině strukturální (z čeho se skládá, podle jakých zákonů, elementů) a funkcionální (jak funguje, k čemu slouží, vývoj, adaptace, procesy).
Základní databázový kurz se skládá z teorie o návrhu databáze, entitně-relačního modelu relačního datového modelu a SQL normalizace. Murray (Murray & Guimaraes, 2008) dále tvrdí, že výuka SQL může mít následující strukturu:

1) úvod do DDL a DML,
2) konstrukce SQL dotazů,
3) animace SQL procedurálním kódem,
4) animace SQL relační algebrou,
5) nesprávné SQL dotazy pomocí relační algebry.

Nad rámec jsou uložené procedury a funkce. Veškeré koncepty SQL musejí být opřeny o teoretickou znalost vnitřního fungování databáze. Z celého komplexu didaktických dovedností učitele předmětů informační výchovy jde hlavně o základní obecnější dovednosti:

1) vytvořit didaktický projekt (scénář, přípravu) na vyučování určitého vzdělávacího obsahu,
2) didaktický projekt transformovat na vyučovací činnosti,
3) analyzovat průběh a výsledky realizace didaktického projektu a výsledky analýzy promítnout do nového projektu nebo do úpravy stávajícího projektu a do jeho nové realizace.

Na potřebu koordinace požadavků praxe a přípravu budoucích učitelů upozornil již v první polovině osmdesátých let Citron (Citron, 1983). Východiskem může být návrh, který popsal Meeneker (Meeker & Nohl, 2007). Povšiml si problému, který se objevuje při výuce databázového kurzu u studentů bakalářského studia. Setkal se s obtíží spočívající v nedostatku času na probrání otázek konkrétních praktických cvičení aplikace databáze. Jako řešení navrhnul do kurzu zařadit několik laboratorních cvičení nebo zadat mini projekt.

3.8 Systematizace obsahu rozvoje databázového myšlení

V této kapitole se zaměřím na obsah rozvoje databázového myšlení. Nástrojem pro rozvoj je vytvoření konceptuálního modelu. Ten je konstruován s ohledem na obsah tématu a hlavní důraz je kladen na vnitřní a vnější souvztažnost aktivit. Vnitřní souvztažnost je tvořena vazbou mezi výukovým cílem, obsahem a metodou. Vnější souvztažnost je tvořena především hierarchickým určením souvislostí jednotlivých komponent. To ve výsledku vytváří podklad pro tvorbu aktivit směřující k rozvoji primárního databázového myšlení. Metody, jež vedou k jeho rozvoji, jsou založeny na procesu hledání prvků, zákonitostí, struktur a principů použitelných pro popis struktury obsahu databázového myšlení. Model tématu by měl být použitelný nejenom při přípravě podkladů pro učivo, ale i při hledání základních dovedností a znalostí, jejich vzájemné hierarchie, posloupnosti a umožňující snížit obtížnost jejich osvojování.

Proces tvorby modelu tématu vychází z didaktické transformace. Skalková (Skalková, 1999) ji popisuje takto: „učivo vzniká zpracováním obsahů představujících různé oblasti kultury (vědy a techniky, umění, činností a hodnot) do školního vzdělávání, tj. do učebních plánů, osnov, učebnic, do vyučovacího procesu. V tom smyslu se hovoří o „didaktické transformaci“. Tu v disertační práci pojímám jako přeměnu obsahů vztahů z pedagogického pole a jejich přetváření. Fakta nejsou přenášena v prosté podobě, např. z vědy, umění nebo praktických činností do školního vyučování. Při tomto zpracování se především uplatňuje kategorie cílů, k nimž je daný obsah zaměřen. Dále je velmi významným činitelem hledisko subjekt žáka. Uvažuje se nejenom o věkových zvláštnostech, ale také o smyslu učiva pro žáka, o roli látky v realitě jeho přítomného i budoucího života. Důležitá je i role učitele. Především jde o to, aby jejich tvůrce sám hluboce chápal podstatu učiva a uměl je zprostředkovat žákům ve vlastním procesu vyučování. Při stanovení obsahu databázového myšlení budu dodržovat následující postup:

1) Nalezení hlavní funkci výuky a nejjednodušší systém, který tuto funkci naplňuje.
2) Identifikace dílčích funkcí pro úplnou realizaci hlavní funkci.
3) Vymezení relací pro vybudování dílčích funkcí.
4) Vymezení nutných prvků výukových situací a jejich odpovídajících znaků.

Z procesního hlediska se systematizací obsahu učiva zabýval Holden (Holden, 2008). Proces hodnocení měří úspěch studentů dosahováním cílů výuky. Studentův úspěch měří mírou dosahování cílů programu a zároveň vrací výsledky zpět do procesu tvorby kurikula.

Holden (Holden, 2008) určil pro výuku databázových aplikací jako stěžejní osvojení následujících kompetencí:

1) návrh relační databáze,
2) implementace relační databáze,
3) dotazování relační databáze.

Jako výsledky učení kurzu vybral následující cíle programu:

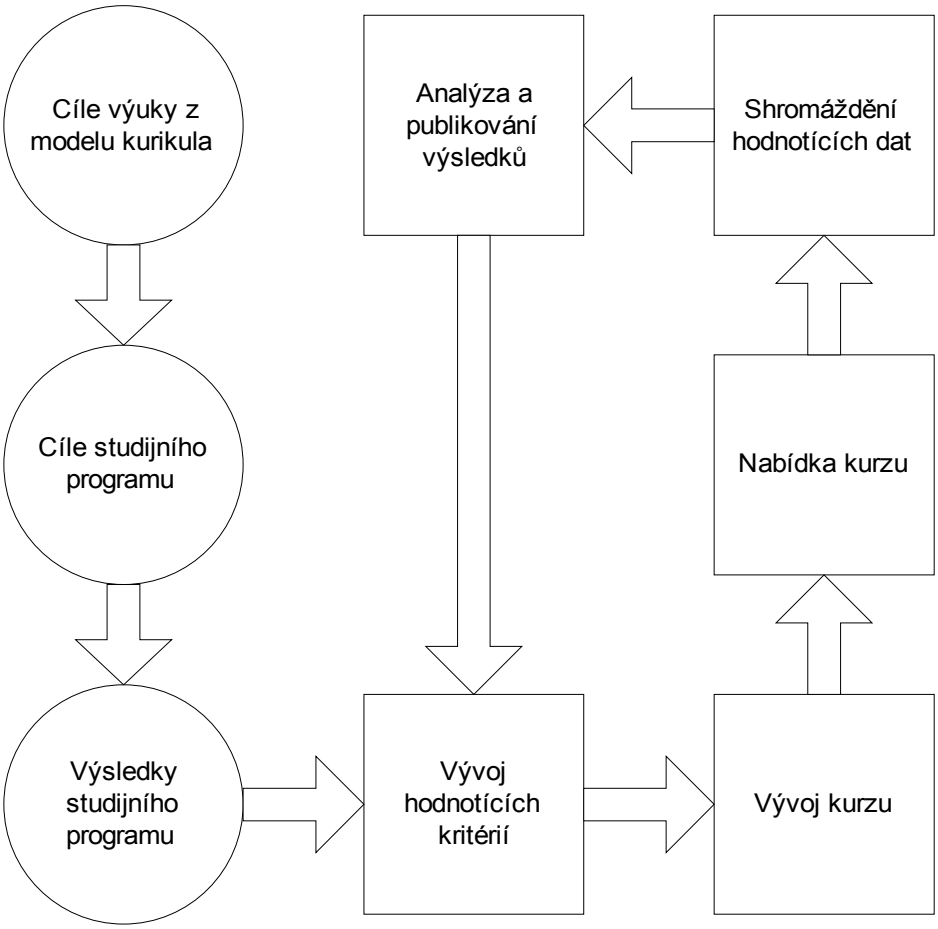
1) schopnost číst a interpretovat diagram ER modelu a mapovat je na relační model,
2) dovednost aplikovat techniky normalizace na relační model,
3) schopnost převést relační model na DBMS produkt pomocí SQL DDL příkazů,
4) dovednost položit dotaz relační databáze pomocí SQL DML příkazů.

Jeho pohled byl zaměřen na dovednosti související s implementací databází. Pro účely své disertační práce jsem ohnisko pozornosti systematizace obsahu posunul na základy dotazovacího jazyka. Ten považuji za hlavní komponentu databázového myšlení. Pro oblast rozvoje dovednosti prezentovat informace jsem stanovil čtyři hlavní etapy, které na sebe navazují tak, jak je dotaz skládán v jednotlivých po sobě jdoucích krocích. Jedná se tedy o procesní pohled na sestavování dotazu.

I. etapa se týká formulace dotazu. Formulace dotazu úzce souvisí s dovedností řešení problému. Skládá se jednak z vnitřního zpracování zadání dotazu a formulace požadavku na výstup dotazu v přirozeném jazyce. Jednak se tedy v této etapě žák pokouší o zasazení zadání do širšího kontextu a dále nalezení prerekvizit a navazujících předmětů. Zároveň si v této fázi může žák dělat poznámky, používat dokumentaci, komunikovat s okolím, rozhodnout se o akceptaci problému k řešení. V oblasti řešení problému potom dekomponuje problém na menší části, hledá zákonitost řešení, používá myšlenkové modely, analyzuje situaci a problém a vybírá nejlepší plán pro řešení problému. Do této etapy zároveň spadá smyslové vnímání úlohy v podobě vnímání zadání, čtení zadání, sluchový příjem zadání a zrakový příjem zadání. Výstupem první etapy je definice problému v přirozeném jazyce.

II. etapa je charakterizována překladem dotazu. Zakládá se na identifikaci a návrhu. Patří do ní využití dovedností z oblasti učení, intuice, grafického zobrazení, tichého čtení, myšlení a analýzy. Součástí je i fáze formulace výběru elementů datového modelu a potřebných operací. Zároveň v ní probíhá interní a externí spojení znalostí. S informacemi jsou prováděny

operace pro organizaci informací, získávání specifických informací a zadávání informací. Do této etapy kognitivního modelu dotazování je nutné zahrnout i osobní preference. Dále jsou uplatněny programovací koncepty. Mezi ty patří syntaktická znalost, sémantická znalost, modularita, strukturované programování, dokumentace, top-down design, odlaďování a externí dokumentace. V rámci aplikace logických pravidel jsou využívány matematické znalosti. Rovněž v této fázi dochází k rozlišení statické a dynamické části. Zároveň sem patří predikce (co se stane, až bude program spuštěn). Patří sem rovněž rozvrstvení jednotlivých sémantických znalostí programovacího jazyka na strojovou, transakční, předpříkazovou, příkazovou, znalostní povinnou a znalostní nepovinnou část, vysokoúrovňové úseky a programové vrstvy. V neposlední řadě zde dochází k uplatnění datového modelu jako znalosti sémantické sítě (ERD), kardinality, práce s relací a entitami. Výstupem druhé etapy je identifikace relací, polí, relačních vztahů a jejich vyjádření v přirozeném jazyce.

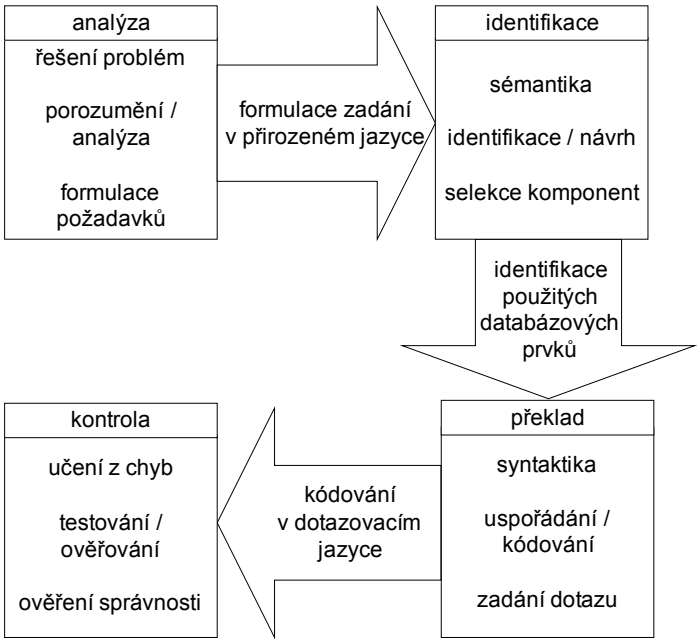


Obrázek 9 Cyklus evaluace kurzu dle Holdena (Holden, 2008)

III. etapa je zaměřena na zadání dotazu. Jedná se o používání syntaktické znalosti za účelem

sestavení konkrétního dotazu v daném dotazovacím jazyce. Rozhodnutí je prezentováno výběrem elementů datového modelu a potřebných operací. V rámci toho se identifikace relací polí, relačních vztahů a jejich vyjádření v přirozeném jazyce přeloží do jazyka dotazovacího. Při překladu do dotazovacího jazyka jsou využívány syntaktické znalosti dotazovacího jazyka. Jedná se o oblast podmiňovací, integritních omezení a datových typů. Podmiňovací oblast sémantické znalosti dotazovacího jazyka se skládá především ze selekce, projekce a řazení. V rámci integritního omezení jsou důležité znalosti primárního klíče, cizího klíče a omezujících podmínek. V rámci oblasti datových typů je důležitá znalost konkrétních implementačních detailů. Tedy například tabulek, sloupců a pohledů. Výstupem třetí etapy je překlad formulace dotazu z přirozeného jazyka do formálního dotazovacího jazyka.

IV. etapa je etapa analýzy odpovědi. V ní se žák dopouští kontroly správnosti a vnímá zpětnou vazbu od systému, do kterého dotaz zadal. Na základě pochopení zpětné vazby dochází k rozvoji databázového myšlení formou učení.



Obrázek 10 Etapy překladu dotazu

Obdobně lze etapy charakterizovat oblastmi myšlenkových operací, které jsou v dané oblasti využívány. V etapě formulace dotazu je to oblast řešení problému, v etapě překladu dotazu se využívá především sémantické znalosti, ve fázi zadávání dotazu jsou hlavní znalosti syntaktické a ve fázi kontroly správnosti zadaného dotazu jsou to hlavně zkušenosti s učením z chyb. Další stránkou databázového myšlení je strukturní pohled na znalost dotazovacího jazyka. Ta se skládá z konstrukce složek jednotlivých znalostních oblastí. Pokud bychom se

podrobněji zabývali syntaktickou znalostí dotazovacího jazyka, lze konstatovat, že se v případě dotazovacího jazyka jedná o analogii struktury syntaktické znalosti programovacího jazyka. Ta obsahuje tři hlavní složky. První je podmiňovací, druhá kontroly toku a třetí obsahuje datové typy. První, podmiňovací je příkazem pro větvení (například if-then-else), druhá pro switch (přepínání) a třetí pro pattern matching, neboli porovnávání řetězce. V rámci kontroly toku se potom objevují jednak primitiva, za druhé smyčky a za třetí strukturované. V rámci datových typů se jedná o rozdělení na statické a dynamické. V rámci sémantické znalosti programovacího jazyka je to strojová úroveň, transakční úroveň, prapříkazová, příkazová, znalost povinných částí, znalost nepovinných částí, nepovinné vysokoúrovňové úseky a programová úroveň. Struktura sémantické znalosti dotazovacího jazyka je analogicky strojová, transakční, strukturní v rámci konceptuálního modelu, entitně-relační a dotazová. Na základě kognitivního modelu programování identifikuji stěžejní principy, bez kterých databázové myšlení nelze osvojit. Z výše uvedených psychologických studií jsem syntetizoval kognitivní model myšlení v dotazovacím jazyce, složený ze čtyř etap databázového myšlení. Pro konstrukci elementů byl tedy zaujat strukturní a procesní pohled na databázové myšlení jako celek, sémantický a syntaktický pohled se zaměřením na dotazovací jazyk. Analogicky lze pro dotazovací jazyk určit rozdělení syntaktických znalostí do oblastí podmiňovací, integritních omezení a datových typů. Do kategorie podmiňovací spadají operace pro projekci, selekci a řazení. Do kategorie integritních omezení lze zařadit fungování principů primárních klíčů, cizích klíčů a omezujících podmínek. V rámci datových typů se potom objevují tabulky, sloupce a pohledy.

Jak uvádí Štípek (Štípek, 2004), je v první fázi procesu vymezení konkrétních elementů třeba nejprve uvažovat o základním přístupu spočívajícím v reflexi principů z hlediska jejich uplatnění při interpretaci. Teoreticky lze v elementech pojmout každý princip izolovaně, nebo naopak zahrnout všechny v rámci elementu jediného. Oba tyto přístupy jsou extrémní, a jak dále ukážeme, ne zcela vyhovující, avšak ohraničují prostor realizace uvedených úvah. „Nelze hovořit o vzájemném vlivu mezi principy, protože k takovému jevu nedochází, ale jejich spolupůsobením rozumíme jejich funkci v procesu interpretace jako izolovaných faktorů, v důsledku určujících výslednou podobu prezentační roviny. Takové pojetí koresponduje s prvním extrémním přístupem k principům jako izolovaným jevům. Jeho přijetí by však v konečném důsledku znamenalo vyloučení reflexe spolupůsobení v rámci elementů, což by se negativně odrazilo v jejich relacích s učivem, které by pak neumožňovalo adekvátně toto spolupůsobení prezentovat (Štípek, 2004).

Druhý extrémní přístup je skutečně pouze hranicí úvah, neboť jeho uplatnění by znamenalo

popření elementarizace a pojem element by ztratil smysl. Z odmítnutí obou extrémních přístupů vyplývá, že soubor elementů bude, zjednodušeně řečeno, souborem určitých množin principů a dalších souvisejících poznatků. V rámci jednoho elementu pak lze prostřednictvím učiva spolupůsobení zohlednit. Nicméně je skutečností, že v procesu interpretace působí společně všechny principy. Je tedy otázkou, jak zajistit reflexi spolupůsobení principů, které nenáleží do stejné množiny, tj. nejsou součástí jednoho elementu. Tato otázka není dosud zodpovězena, a proto představuje problém, jejž je třeba řešit ještě před určením konkrétních elementů."

Navržené klíčové prvky lze chápat jako znalostní principy, jejichž zvládnutí je nezbytné pro rozvoj databázového myšlení. Jednotlivé prvky jsou:

1) hierarchie jako princip aplikovaný v oblasti souvislosti jednotlivých abstraktních prvků databázového myšlení, tj. vztah, kardinalita, atribut a datový prvek,
2) entita jako kategorie obrazu předmětů do databázové struktury. Při jejím použití je nezbytná dovednost určit skupinu předmětů, jež spojuje určitá vlastnost. Na jejím základě je stanovena abstraktní skupina neboli entita,
3) výběr je myšlenková operace umožňující v rámci skupiny předmětů vybrat pouze ty odpovídající vlastnosti definované kritérii,
4) uspořádání je myšlenková operace umožňující předměty uspořádat podle vybraného kritéria tak, že se jednotlivé prvky množiny porovnávají a podle daného znaku se sestaví jejich uspořádání neboli pořadí,
5) seskupení je myšlenková operace umožňující spojit skupinu předmětů do určité kategorie na základě shodného znaku nebo vlastnosti těch předmětů, u kterých daný znak nabývá stejné hodnoty.

Při systematizaci obsahu databázového myšlení je stále nezbytné mít na paměti budoucí účel vzniku učiva pomocí didaktické transformace. V rámci přípravy lze stanovit kategorie cílů, k nimž je daný obsah zaměřen. Použité metody se opírají o disertační práci (Štípek, 2004), ve které Štípek didakticky analyzoval značkovací jazyky. Účelem analýzy byla formulace teoretického způsobu pojetí obecného tématu jako východiska pro další zpracování didaktickou transformací tak, aby adekvátně reprezentovala hlavní vztahy a principy daného tématu. Značkovací jazyky analyzoval ve dvou fázích. V první fázi se zabýval formulací obecného modelu a ve druhé tvorbou modelu tématu. Podle Štípka (Štípek, 2004) má být obecný model aplikovatelný na širší spektrum tematických celků. Prvním krokem při

konstrukci obecného modelu je analýza tematického pole jako základního učiva, jež si má žák osvojit. Úplné pole obsahuje všechny tematické okruhy základního učiva, ale nelze je chápat jako pouhý výčet názvů nebo seznam témat. Zahrnuje i vztahy, souvislosti, strukturu a principy uvnitř prvků pole nebo mezi nimi.

1) slovník pojmů,
2) hierarchie pojmů,
3) struktura pojmů,
4) procesní zařazení pojmů,
5) kompetence pro manipulaci prvky (pojmy),
6) množina nejnutnějších pojmů,
7) množina volitelných pojmů,
8) úrovně abstrakcí pojmů.

V rámci toho existuje tzv. dílčí datová struktura, neboli entita, která slouží jako struktura pro uložení informace. V rámci definice slovníku pojmů jsem vytyčil minimální skupinu pojmů nutných pro pochopení databázového myšlení.

ad 1) Slovník pojmů. První skupina pojmů obsahuje prvky: tabulka, sloupec, řádek, buňka, index, primární klíč, cizí klíč, relace, kardinalita, datový typ, integritní omezení. Druhá skupina obsahuje operace: vložení záznamu, odstranění záznamu, aktualizace záznamu, selekce, projekce.

ad 2) Hierarchie pojmů je následující:

Tabulka je nadřazena pojmům řádek, sloupec.
Řádek a sloupec jsou nadřazeny pojmu buňka.
Index je vlastnost sloupce.
Primární klíč je vlastnost sloupce.
Cizí klíč je vlastnost sloupce.
Selekce je operace nad tabulkou.
Projekce je operace nad tabulkou.
Relace je vztah mezi tabulkami.
Kardinalita je vlastnost relace.
Datový typ a integritní omezení jsou vlastnosti sloupce.

ad 3) Struktura pojmů je následující: Tabulka se skládá ze sloupců. Do tabulky může být vložen řádek, jehož struktura je určena právě sloupci tabulky. Každý průsečík mezi sloupcem a řádkem se nazývá buňka. V rámci zrychlení přístupu k datům může mít sloupec vytvořený index pro rychlejší přístup k údajům. Zvláštní druh sloupce je primární klíč, který musí splňovat podmínky jedinečnosti. Cizí klíč je potom klíč, který je do tabulky přiřazený z jiné tabulky na základě relace. Relace je vztah mezi dvěma nebo více tabulkami a umožňuje spojení údajů z různých tabulek. Selekce je potom výběr pouze určitých řádků tabulky. Projekce je výběr pouze určitých sloupců tabulky. Kardinalita je vlastnost relace, která umožňuje určit, kolik řádků jedné tabulky odpovídá počtu řádek z druhé tabulky. Datový typ je vlastnost sloupce a určuje, jaký typ dat může být v daném sloupci uložen. Integritní omezení je vlastnost sloupce, která umožňuje specifikovat pravidla pro vložené údaje při jejich vkládání.

ad 4) Procesní zařazení pojmů: V rámci procesu je první v pořadí vytvoření tabulky. To může být provedeno, pouze pokud je známa specifikace sloupců, ve druhé řadě lze specifikovat typ sloupce, zdali se jedná o primární klíč, zdali má mít index, zdali má mít primární klíč jako funkční jednotku pro práci s údaji v tabulce, zdali má vztah k jiné tabulce, jaký typ informací se bude do sloupce ukládat a jaká integritní omezení má mít sloupec.

ad 5) Kompetenční pohled: Jednotlivé koncepty jsou uchopeny jako jednotlivé prvky. Koncept tabulky je založen na klasické tabulce tvořené sloupci a řádky. V rámci databázového myšlení je její reprezentace složena ze sloupců a řádků. Důležitá je schopnost stanovit vhodné kritérium pro výběr vhodných řádků. Způsob rozdělení údajů do jednotlivých tabulek je založen na seskupení předmětů do skupin. Pojmenování jednotlivých sloupců je založeno na vlastnostech objektu. Jedná se tedy v tomto smyslu o reprezentaci objektů reálného světa a jejich vzájemných vazeb. Vytváření indexu je z kompetenčního pohledu založeno na schopnosti určit, který sloupec ve které tabulce bude používán nejčastěji. Relace, primární a cizí klíče jsou vyjádřením schopnosti převést vazby reálného světa do jejich vzájemných relací. Pro použití projekce a selekce je zapotřebí schopnost určit a formulovat, jaká data je nutné získat z tabulky. Buňka je základní informační jednotkou, elementem, se kterým se pracuje. Určení datového typu je založeno na sledování reálného světa, pochopení atributu a zároveň určení vhodného typu záznamu v tabulce. Integritní omezení spadá do oblasti technických kompetencí a je založeno na stanovení možných hodnot, které mohou být

do sloupce zadávány. Při jeho určení je nezbytné provést rozhodnutí o tom, jaká data lze, a nelze do databáze ukládat.

ad 6) Při bližší analýze nejdůležitějších pojmů rozvoje databázového myšlení jsem dospěl k vymezení tří nejdůležitějších pojmů: tabulka, projekce a selekce.

ad 7) Volitelné pojmy jsou oproti tomu rozsáhlejší množinou a patří do nich všechny pojmy, které se týkají implementačních detailů. Jako takové jsou to tedy především pojmy index, integritní omezení, primární a cizí klíč.

ad 8) Úrovně abstrakce vycházejí již ze samotného pojetí databází, které odděluje uživatele od implementačních detailů. Na základě toho jsou tedy základními abstraktními rovinami rovina entitní, relační, atributní a fyzická. Do entitní jsou zahrnuty pouze jména entit. V relační jsou jména entit a vztahy mezi nimi. V množině atributní jsou jména entit, relace mezi entitami a jména atributů. Do fyzické úrovně abstrakce spadají jména tabulek, relace mezi nimi, jména sloupců a datové typy sloupců včetně jejich integritních omezeními.

Dle stanovených cílů práce je má rozpracování modelu rozvoje databázového myšlení přispět etablující se didaktice informačních výchovy. Výchozím postojem je názor, že existují dva hlavní strukturní znalostní elementy, a to deklarativní a procedurální. Deklarativní strukturní element zahrnuje především fakta, která lze jasně definovat a vymezit. Procedurální elementy jsou především naučené dovednosti, které slouží primárně k aplikaci elementů deklarativních. Aby bylo možné ponořit se až k podstatě databázového myšlení a zkoumat tyto základní principy z pohledu struktury obsahu, je nutné nejdříve přednést obsah databázového myšlení jako takového z pohledu pedagogiky, psychologie a informatiky. Pohled odborný, pohled informatiky v tomto případě zastupuje procesní pohled na rozvoj kompetencí databázového myšlení. Základem vymezení pole elementů databázového myšlení je systematické rozpracování teorie databázových systémů, konceptuálního modelu a dotazovacího jazyka s cílem ujasnit používanou terminologii za účelem nalezení základních stavebních kamenů podstaty databázového myšlení. K tomu, aby mohla být elementaristika databázového myšlení vhodně rozpracována, je tedy nezbytné přistupovat k ní z širšího pojetí a do elementů zakomponovat i aspekt rozvoje dílčí kompetence dotazovacího jazyka. Kromě toho vzít v úvahu kognitivní model znalosti dotazovacího jazyka jako klíčové komponenty databázového myšlení.

Aby bylo možné do konceptu informační výchovy zahrnout koncept rozvoje databázového myšlení jako integrální součást informační gramotnosti, byla provedena identifikace základních prvků databázového myšlení. Jejich rozpracování může přispět k rozvoji didaktiky etablující se informační výchovy tím, že systematicky rozpracovanou strukturu rozvoje databázového myšlení aplikuje do aktivit, na jejichž základě lze potom didaktickou transformací vytvořit učivo určené pro primární setkání s databázovým myšlením. Vzhledem k tomu, že veškeré snažení v oblasti strukturace databázového myšlení vychází ze základů pro sestavování učiva pro žáky, kteří prožívají primární poznání databázového myšlení, je následující kapitola věnována otázce praktického využití systematizace databázového myšlení. Základem pro jejich konstrukci je rozbor procesní stránky databázového myšlení. Z poznatků o chybovosti při učení dotazovacího jazyka lze stanovit problematické oblasti, na které bude nezbytné se zaměřit při předávání znalostí o databázovém myšlení. Naopak je v rámci tohoto konceptu žádoucí posílení rozpadu elementů na menší, lépe pochopitelné celky.

4 Vyučovací metody utváření a rozvoj databázového myšlení

4.1 Východiska psychologie programovacího jazyka

Jedním z autorů publikujících v oblasti psychologie programovacích jazyků je profesor psychologie na Kalifornské univerzitě, Richard E. Mayer. Mayer ve svých výzkumech v sedmdesátých a osmdesátých letech dvacátého století sledoval vliv vyučovacích metod na kognitivní strukturu výsledných znalostí. Experimenty prováděl nejenom ve výuce aplikované matematiky (Mayer & Greeno, 1972), (Mayer, 1974), (Mayer, 1975), ale i ve výuce programovacích jazyků (Mayer, 1979), (Mayer, 1981). Pro oblast rozvoje databázového myšlení považuji za zásadní poznatky v oblastech komparace strukturálních rozdílů výsledků učení programování a kognitivní modely znalosti programovacích jazyků. Ty v disertační práci slouží jako myšlenkový podklad pro systematizaci databázového myšlení, konstrukci procesu dotazování a sledování původu chyb vznikajících při učení dotazovacího jazyka.

První výzkumy, na jejichž základě byly postaveny modely znalosti programovacího jazyka, prováděl Mayer (Mayer & Greeno, 1972), když zkoumal strukturální rozdíl výsledné znalostní struktury studentů při aplikaci rozdílných vyučovacích metod. Pomocí experimentů zkoumal vliv vyučovací metody na kognitivní strukturu výsledku učení. Srovnával metody zaměřené na procvičování výpočtů vzorců a metody objasňující vazby mezi proměnnými ve vzorci. Charakteristiku kognitivní struktury znalostí studenta zjišťoval testem jeho znalostí o vzorci. Výsledek pokusu potvrdil strukturální rozdíl ve výsledných znalostech studentů. Mayer se domníval, že strukturální rozdíl spočívá v různých typech vazeb mezi znalostmi. Identifikoval mezi nimi dva základní typy. První nazval vnitřní spojení, pro něj je charakteristická silná vazba znalostí mezi sebou navzájem. Druhý typ pojmenoval vnější spojení, to je charakteristické silnou vazbu na ostatní kognitivní struktury znalostních prvků studenta. Jeho zjištění má v rámci teoretického zpracování databázového myšlení zásadní dopad na poznatky o procesním modelu dotazování. Může tak sloužit jako podnět pro odlišení rozvoje vnitřního spojení databázového myšlení jako uzavřeného teoretického konceptu od procesu rozvoje vnějšího spojení struktury poznání databázového myšlení na ostatní studentovy znalostní struktury.

Další významný Mayerův článek (Mayer, 1974) z roku 1974 má název: „Proces rozvoje procesů a odolnosti procedur řešení problémů vlivem změn podmínek při jejich testování“. Výsledky experimentu opět prokázal odlišný výsledek učení při použití různých vyučovacích

metod. První skupina studentů byla podrobena výuce zaměřené na procvičování výpočtu pomocí vzorce. Výuka ve druhé skupině kladla důraz na vysvětlení významu vzorce. Experiment byl proveden na 117 studentech, jejich úkolem bylo odpovědět na 30 otázek. Každá odpověď studenta byla klasifikována do kategorie správná/nesprávná. Studenti v kontrolní skupině dosáhli méně než 10 % správných odpovědí. Výsledky všech studentů analyzoval Mayer metodou analýzy rozptylu. Zjistil tak rozdíl ve výsledných znalostech obou skupin především v oblasti tzv. blízkého a vzdáleného přenosu. Studenti, u kterých byl prokázán lepší blízký přenos, zaznamenali lepší výsledky při testování znalostí v oblasti vnitřního spojení kognitivní struktury znalostí. Oproti tomu skupina studentů s lepším vzdáleným přenosem zaznamenala lepší výsledky při testování znalostí v oblasti vnějšího spojení kognitivní struktury znalostí. Mayer zároveň interpretoval výsledky tak, že zvýšení počtu lekcí má pozitivní vliv na oba druhy znalostí (vnitřní i vnější spojení).

Pro teoretický koncept primárního rozvoje databázového myšlení jsou zásadní výsledky experimentu z hlediska poznatků v oblasti originálního řešení problémů. Vztaženo k povaze primárního rozvoje databázového myšlení, jenž je charakteristický nutností zvládnutí originálních a tvůrčích činností, z toho vyplývá nutnost vybudovat stabilní kognitivní strukturu s množstvím vnějších spojení. Tím je vytvořen předpoklad pro rozvoj vzdáleného přenosu mezi kognitivní strukturou databázového myšlení a ostatními znalostmi. V praxi pak může mít teoreticky lepší výsledky při rozvoji primárního databázového myšlení aplikace výukových metod zaměřená na analýzu formulovaných dotazů.

Další Mayerův experiment (Mayer, 1975) byl zaměřen na rozvoj znalosti počítačového jazyka. Ve svém výzkumu podrobil 176 počítačových laiků výuce jednoduchého programovacího jazyka. Přitom první část studentů podrobil výuce zaměřené na interpretaci programů a u druhé části studentů poskytnul podrobné vysvětlení principu fungování počítače analogií s věcmi běžného života. Po skončení výuky porovnal rozdíl ve výsledné kognitivní struktuře znalosti studentů. Experiment prokázal u studentů vyučovaných metodou procvičování interpretace programů lepší výsledky při řešení iteračních problémů. Ti studenti, kteří absolvovali výuku bez důrazu na kognitivní model počítače, dosáhli lepší výkon v oblasti přímočarého sestavování programů. Kognitivní model počítače byl důležitý především pro studenty bez rozsáhlejších odborných znalostí. Procvičování interpretace programů bylo nejprospěšnější pro studenty s výukou bez modelu počítače. Druhá část experimentu měla tři fáze. V první fázi se zjišťoval vliv charakteru vyučované látky na výsledek učení. Pro vysvětlení principů počítače používal analogii s předměty běžného života, jako jsou světelné tabule, pokladny nebo nákupní seznamy. K vysvětlení vývojových

diagramů použil odbornější termíny a analogie s geometrickými symboly a grafy.

Druhé fáze Mayerova experimentu se v rámci úvodního kurzu psychologie účastnilo 80 studentů univerzity. Výsledky každého studenta zaznamenal do faktorové matice. První faktor obsahoval údaje o použité metodě výuky (pravidlo, model, model-tok, pravidlo-tok, model-tok), druhý faktor množství procvičované látky (procvičoval nebo neprocvičoval). Všichni studenti absolvovali srovnání shodným post-testem. Studentům vyložil jednoduchý programovací jazyk. Jako podpůrný prostředek použil dva typy příruček. První typ příručky obsahoval definici sedmi příkazů a druhý příklady založené na různých myšlenkových modelech. Post-testový dotazník se skládal z osmnácti položek. Pro porovnání výkonů studentů použil první kritérium typu zpracování problému, a to buď generování, nebo interpretace. Druhé kritérium odráželo složitost problému a obsahovalo kategorie: „jednořádkový program (příkaz)“, „program bez iterací“ a „program s iterací“.

Předložený studijní materiál zahrnoval úvodní dotazník zjišťující předchozí zkušenosti studentů s programováním. Studenti byli rozděleni do malých skupin po dvou až čtyřech podle typu výuky. Nejprve vyplnili vstupní test a krátký dotazník, do kterého uvedli kvantitu svých znalostí počítačového programování. Dále jim byly rozdány příručky a měli za úkol nastudovat látku v nich obsaženou tak, aby „porozuměli jejím obsahu“, a tím se připravili na „test“. Když dokončili čtení, dostali osm příkladů a formulář pro odpovědi. Každý student odpověděl na každou otázku samostatně a ihned obdržel zpětnou vazbu ve formě správné odpovědi na rubu karty. Když student dokončil procvičování, byl požádán o účast na post-testu. Studenti mohli pracovat svým tempem, ale odpovídali pouze na jednu otázku současně, bez možnosti vracet se k předchozím kartám. Úlohy byly seskupeny podle obtížnosti. Po opakovaném přečtení instrukcí na konci experimentu měl student udat míru ztotožnění s předloženými koncepty. Výsledky pěti studentů, kteří měli předchozí zkušenost s programováním, byly vyřazeny a místo nich byly zařazeny výsledky jiných studentů. Každá odpověď byla vyhodnocena podle správnosti.

Mayer provedl analýzu post-testu použitím rozptylové analýzy. Skupina studentů podrobených výuce zaměřené na model excelovala v úlohách náročných na interpretaci. Skupina, která neměla při učení k dispozici model počítače, excelovala v kreativních úlohách. Kategorie a komplexita problému se u obou skupin shodovaly, ale skupina vyučovaná za podpory modelu řešila lépe problémy náročné na interpretaci, vytváření a vysvětlování iterací. Skupina bez modelu byla nejlepší v přímých problémech a neopakujících se generativních položkách.

Výsledek ukazuje, že výuka s podporou modelu vyústila v kvalitativně jiný výsledek učení

než výuka bez modelu. Porovnáním skupin vyučovaných za pomoci vývojových diagramů a bez nich dokázal obousměrnou interakci zahrnující výukovou metodu a typ problému. Výsledek také ukázal, že interakce zahrnující vývojové diagramy vyústily v slabší výkon přenosu znalostí potřebujících rozšířenou aplikaci materiálu na nové situace (např. interpretaci nebo opakování) a relativně lepší výkon na položkách, které potřebují aplikaci myšlenek uvedených ve výukových materiálech. Je patrné, že vývojové diagramy podporují jenom úzkou část předchozích zkušeností s geometrickými symboly a neposkytují širší množinu zkušeností, které mohou být použity ke kódování nové informace. Z druhého experimentu vyplynulo zjištění o odlišnosti výsledků strukturálního a kvalitativního učení. Mayer poukázal na tendenci pozitivního vlivu použití modelu u slabších studentů, ale snížení výkonu u nejlepších studentů. Z celého experimentu je pro rozvoj databázového myšlení důležitý poznatek strukturálního rozdílu výsledné kognitivní struktury, jenž může být zobecněn a použit při rozhodování, zdali do výuky začlenit entitně-relační diagramy.

Mayerovy výzkumy patří do kategorie výzkumů psychologie programování. Stejnou oblastí se zabýval Weinberg (Weinberg, 1985). Ve své knize „The Psychology of Computer Programming“ vymezuje programování jako výkon člověka, společenskou aktivitu a aktivitu jednotlivce. I v ostatních psychologických studiích je programování často pojímáno jako výkon člověka. Jak vyplývá mimo jiné i z Weinbergovy studie, může být výkon žáka, jako každý jiný výkon, měřen počtem chyb, kterých se při formulace dotazu dopouští. Na základě stanovení měřitelného kritéria rozvoje databázového myšlení v oblasti dotazovacího jazyka vyplývá možnost používat výkonová měřítka a na jejich základě řídit proces výuky. Výše uvedený teoretický koncept je často základním stavebním kamenem při vývoji prostředků pro podporu rozvoje databázového myšlení.

Další oblastí Weinbergovy studie je oblast analýzy zdroje chybovosti při formulaci kódu v počítačovém jazyce. Vychází z přesvědčení, že na počítačový program je nezbytné pohlížet nejenom jako na instrukci pro práci počítače, ale zároveň jako na psaný projev, jenž bude v budoucnosti číst nebo upravovat člověk. Z toho poznatku vycházejí teoretické koncepty rozvoje počítačových jazyků. Při jejich rozvoji tedy autoři berou ohled nejenom na omezení stroje, ale i na limity myšlení člověka. Z výše uvedeného konceptu pochází teoretická klasifikace chyb, jež se mohou vyskytnout při učení počítačového jazyka. Základním rozdělením je tak klasifikace chyb vyplývajících z neznalosti programovacího jazyka nebo nedostatku zkušeností. Neznalost programovací jazyka se neobjevuje pouze ve formě neznalosti syntaxe. Jinou podobu může nabýt, pokud žák nezná teoretické principy databázového myšlení nebo není schopen uchopit velkou část myšlenkového procesu

formulace dotazu. Výše uvedené poznatky jsou podstatné především při diagnostice zdrojů chyb při učení. Zároveň mohou být teoretickým základem pro vývoj analytických postupů pro optimalizaci výukových metod.

Saj-Nicole (Saj-Nicole et al., 1983) ke zdrojům chybovostí studentů při učení programovacích jazyků dodává, že znalost zdrojů běžných chyb při formulaci programových konstruktů je klíčovou součástí znalostních kompetencí studenta. Při vedení kurzů programování podrobila podrobné analýze zdrojové kódy studentů. Zaměřila se na analýzu chyb v jejich programech. Výsledek analýzy ukázal, že do učiva v oblasti programovacích jazyků by bylo přínosné zahrnout kromě výuky syntaxe a sémantiky programovacího jazyka i příklady běžných chyb, kterých se studenti dopouštějí. Chyby studentů rozdělila do dvou hlavních kategorií. Do první kategorie zahrnula chyby způsobené nesprávnou reprezentací stereotypních sekvencí akcí. Do druhé kategorie zařadila chyby způsobené nekorektním převedením zadání do srozumitelného a vykonatelného programu. Chyby tedy klasifikovala na chyby zapříčiněné programátorem a chyby způsobené nesprávným porozuměním zadání. Chyby zaviněné programátorem dále rozdělila chyby patrné a skryté. Skryté chyby student neobjevil hned, ale zjistí je až při běhu programu. Do první kategorie chyb plynoucí z nesprávného pochopení zadání zařadila chyby plynoucí z nesrovnalosti mezi znalostmi studentů a obsahem výuky. Do druhé kategorie zařadila chyby plynoucí z nesprávného nebo nedostatečného pochopení programovacích principů. Třetí zdroj chyb určila jako nepřiměřené zobecnění. To se objevovalo v případě nesprávného pochopení kontextu zadání. Do čtvrté skupiny chyb zařadila nesprávné pochopení konceptuálního významu programovacího konstruktu a jako pátý zdroj chyb určila neznalost pravidel programovacího jazyka.

Další vědec působící v oblasti rozvoje programování je Biffah (Biffah et al., 1997). Ten se pokusil najít a rozkrýt kognitivní reprezentaci datové struktury v primárním střetu s praktickou výukou programování. Primární střet s programováním specifikoval jako první setkání s programováním u začínajících studentů. Svoji teorii založil na domněnce, že efektivní použití různých vizuálních reprezentací aplikovaných v rámci domény lineárních datových struktur pole, zásobníku, fronty a spojeného seznamu přispěje k lepšímu pochopení principů teorie programování. Pro svůj výzkum použil systém MRUDS (Multiple Representation for Understanding Data Structure). Ten obsahoval tři hlavní moduly výuky. Jednotlivé modely používaly výukové principy založené na analogii, reprezentaci a algoritmu. Biffah se domníval, že obrazové, grafické a diagramové podpůrné prostředky pomáhají osvojení znalosti datových struktur a rozvoji strukturální reprezentace základních programovacích konceptů. Ve svém článku se zaměřil na evaluaci počítačového výukového

systému, který založil na formativní a sumační evaluaci. Výsledky formativní evaluace pomohly definovat a zlepšit cíle a metody návrhu. Pomocí sumační evaluace zajistil růst efektivity vzdělávacího produktu v rámci jeho provozu. Výsledky formativní evaluace byly použity pro vývoj systému, což mu umožnilo zaměřit se na zdroje nesprávného úsudek, pomohly mu předejít plýtvání časem a zaměřit pozornost na podrobnosti efektů učení. Poznatky, které vyplynuly z Biffahova výzkumu, potvrzují domněnku o zvýšení průměrné efektivity osvojování programovacích při použití vizuálních podpůrných nástrojů.

Jiné výzkumy psychologie programování jsou založeny na poznatcích z oblasti teoretických konceptů rozvoje efektivity programovacího jazyka. Tomek (Tomek, 1982) porovnával kritéria pro evaluaci programovacích jazyků. Stanovil faktory s vlivem na atraktivitu programovacího jazyka u začínajících programátorů. Zjistil, že je determinována mírou rozvoje originálního způsobu řešení problému. Svoje tvrzení doložil výsledky výuky programovacího jazyka Turtle. Domníval se, že pokud je jazyk primárně vyvinut pro potřeby výuky programování, měl by být jakousi hračkou a není u něj rozhodující bezpečnost nebo rychlost. Tomek při klasifikaci programovacích jazyků srovnával jejich vzdálenost od klasických programovacích jazyků. Jeho poznatky jsou stěžejní při hledání výukových metod zvyšujících motivaci studentů. Vyplývá z toho, že netradiční výukový nástroj rozvíjející primární rozvoj databázového myšlení může vést k vyššímu zájmu studentů. Zároveň otevírá otázku, jaké vlastností má mít program, jehož hlavní náplní je být výukovým nástrojem.

Vliv uživatelského rozhraní na práci uživatele zkoumal Chan (Chan et al., 1997). Porovnávalo rozdíl v chování uživatele při použití textového a grafického uživatelského rozhraní pro práci s datovou bází. Hlavní hodnotící kritéria byly přesnost uživatelovi práce, jistota uživatele a rychlost práce uživatele. Výzkumné pole tak založil na čtyřech hlavních faktorech. Prvním faktorem je uživatel, druhým faktorem je datový model, třetím faktorem je typ řešené úlohy a čtvrtým faktorem je určitá charakteristika systému. Nezávislé proměnné, jež vstupují do modelu výzkumného pole, jsou datový model, typ úlohy a charakteristika systému. Determinantu výkonu žáka stanovil jako závislou proměnnou měřenou mírou sebedůvěry žáka, přesností a rychlostí jeho odpovědí. Přesnost položeného dotazu měřil ve škále od nuly do pěti. Každý studentův dotaz nechal vyhodnotit dvěma nezávislými hodnotiteli a pro každý odlišit dvě odlišné stupnice. První stupnice hodnotila sémantickou správnost a druhá syntaktickou chybovost. Rychlost žákovi odpovědi měřil od okamžiku, kdy byla otázky zobrazena na obrazovce do momentu, kdy žák potvrdil správnost zadané odpovědi. Po každé odpovědi se systém dotázal žáka na subjektivní míru sebedůvěry k odpovědi. Vyplnění dotazníku předcházela výuka za pomoci učebnice. V rámci experimentu byly sledovány

vybrané uživatelské charakteristiky. Sledoval inteligenci, zkušenost s programováním a schopnost koncentrovat se po dobu trvání experimentu. V experimentu se zaměřil na faktory související s chápáním principů databáze. Uživatelské charakteristiky byly kontrolovány rozdělením subjektů do experimentálních skupin.

Systémové charakteristiky chápal Chan jako fyzické aspekty systému. První systémová charakteristika byla kapacita měřená dobou odezvy u jednotlivých vstupně-výstupních zařízení. Dalším aspektem byl styl grafického rozhraní a podoba uspořádání prvků uživatelského rozhraní. Pro účely experimentu vyvinul speciální grafické a textové uživatelské rozhraní. Textové rozhraní mělo podobu integrovaného textového editoru s pokročilými funkcemi pro zobrazení dotazů a záznamu experimentálních měření. Vizuální rozhraní zahrnovalo grafické zobrazení integrovaných funkcí pro manipulaci a záznam práce se systémem. Rozhraní nebylo připojeno k produkčnímu databázovému systému, aby Chan předešel ovlivnění výsledků externím systémem. Toho se vyvaroval, neboť přepokládal, že pokud připojí systém na externí databázový systém, dojde ke zpoždění, jež bude způsobené vlivem delší odezvy systému. Kromě toho chtěl zamezit nárůstu počtu chybových zpráv. Po skončení experimentu provedl analýzu dotazů studentů. Nejdříve provedl jejich rozdělení na jednoduché a složité. Kategorii jednoduchých dotazů specifikoval jako dotazy, které pracují s jednou entitou nebo relací. Do skupiny složitých dotazů zahrnul dotazy, jež pracovaly s více než jednou entitou. Výsledky jeho výzkumu jsou cenným zdrojem pro rozhodování, zdali ve výuce vedoucí k rozvoji databázového myšlení použít textové nebo grafické rozhraní.

Underwood (Underwood et al., 1990) zkoumal možnosti použití relačních databází ve výuce výzkumu anglické historie. Účastníkům experimentu ve třech lekcích představil způsob, kterým mohli vyhledávat v databázi. Po ukončení experimentu provedl analýzu souvislostí mezi typem uživatelů a jejich pokusy o vyhledávání v databázi. Výsledky výzkumu poukázaly na vliv předchozích zkušeností s počítači a znalosti historie na efektivitu získávání dat. Underwood ke svým závěrům dospěl především na základě počtu správně zodpovězených dotazů. Druhou charakteristikou potvrzující jeho zjištění byl i čas, který student potřeboval k vyřešení úlohy. Underwood předpokládal vliv základních dovedností na porozumění struktury databáze. Výsledek analýzy první lekce ukázal pozitivní vliv předchozí zkušenosti práce s počítačem na efektivitu vyhledávání v databázi. Výsledek analýzy druhého sezení vliv zkušeností práce s počítačem na efektivitu analýzy nepotvrdil, ale jev se projevil na posledním sezení. Underwood však kromě toho identifikoval ještě další dva faktory, jež můžou mít vliv na efektivitu vyhledávání v databázi. První faktor je schopnost zacházet s klávesnicí. Druhý faktor může být ochota uživatelů aktivně pracovat s nabídkami

a nápovědou databázového systému. Oproti tomu odborná znalost dějin neměla na výsledky experimentu prokazatelný vliv. Jedná se o jeden z odborných výzkumů práce uživatele s databází. Pro zkoumání faktorů rozvoje databázového myšlení jsou tak výše uvedené faktory přínosné především zjištění v souvislosti s faktory, jež mohou u laika vést k úspěšnému řešení zadané úlohy.

Wilson (Wilson, 1987) se zaměřil na konceptuální přístup při výuce databázového myšlení. Ve svém výzkumu popsal problémy spojené s výukou datového modelování. Zjistil, že studenti mají tendenci používat při modelování syntaktickou strukturu vět, která vyplývá z jejich uvažování v přirozeném jazyce. Svoji domněnku se pokusil ověřit analýzou chyb studentů vznikajících při tvorbě konceptuálních modelů databáze. Došel k názoru, že pokud student formuluje dotaz, může mít opravdu překlad dotazu ze zadání v přirozeném jazyce vliv na druh chyb vyskytujících se v dotazovacím jazyce. Chyby se objevily především v oblastech chybné kardinality relací, určování nebinárních relací, nesprávně identifikovaných entit, záměn entit a vztahů a v podobě syntaktických chyb. Výsledek analýzy ukázal, že zdroj záměn entit a vztahů byl způsoben jejich pojmenováním v přirozeném jazyce. Jeho výzkum je podstatný pro oblast konceptuálního modelování a procesu řešení problému studentem. Ukazuje na souvislost překladu zadání z přirozeného jazyka a jeho interpretace v datovém modelu. Pro rozvoj databázového myšlení může být poznatek užitečný především při hledání zdroje chyb v oblasti rozvoje konceptuálního modelování databázové struktury.

Schlager (Schlager & Ogden, 1986) měl vlastní představu o kognitivním modelu myšlení formulace dotazu v dotazovacím jazyce. Motivací pro konstrukci modelu mu byla potřeba zmapovat a zaznamenat vliv použitých učebnic na výsledek učení a stanovení jejich role ve vyučování. Jeho cílem bylo učinit dokumentaci pro běžného uživatele srozumitelnější. Za tím účelem provedl dva experimenty. V nich sledoval vliv výběru dokumentace na výsledek učení programovacího jazyka. V prvním experimentu použil dokumentaci, jež zahrnovala praktické rady. Ty získal rozhovory s profesionály na téma osvědčených praktik.

Ve druhém experimentu do dokumentace zahrnul kognitivní model učení znalosti dotazovacího jazyka. Kromě toho přidal ještě další informace týkající se principu fungování dotazovacího jazyka. Tři skupiny začínajících programátorů bez praxe nechal prostudovat tři různé varianty výukových materiálů. Jednotlivé druhy dokumentace obsahovaly buď procedurální informace, konceptuální informace anebo oba druhy informací. Jako objekt zkoumání stanovil právě dotazovací jazyk. Schlager jej vybral, protože považoval dotazovací jazyk pro podobné experimenty za nejvhodnější. Jeho názor vyplýval z poznání jazykové struktury jazyka. Ten má složitou konceptuální strukturu a relativně malý počet klíčových

slov. V rámci experimentu provedl Schlanger analýzu záznamů rozhovorů. Jejich výsledek umožnily Schlangerovi klasifikovat kognitivní operace použité při dotazování do tří kategorií. První kategorie zahrnovala konceptuální, druhá procedurální a třetí heuristickou. Konceptuální kognitivní procesy zahrnovaly popis principu fungování dotazovacího jazyka a princip provádění dotazu systémem řízení báze dat. Procedurální informace se skládaly z popisu sekvence akcí, jež byly provedeny nad databází, znalost metod vykonání dotazu v databázi a kognitivní myšlenkové operace související s procesem formulace dotazu v dotazovacím jazyce. Heuristické informace popisovaly logická pravidla fungování dotazovacího jazyka. U nich byl kladen hlavní důraz na vlastnosti specifické pro dotazovací jazyk. Kognitivní model znalosti dotazovacího jazyka klade důraz na konzistenci a práci se systémem. Když Schlanger porovnával výsledky obou experimentů, došel k závěru, že v případech, kdy zahrnul kognitivní model jako expertní znalost v podobě nezávislé na produktu, došlo k pozitivnímu ovlivnění výsledků učení začátečníků. Jeho závěry jsou z hlediska mé disertační práce obecně platné v oblasti databázového myšlení především v podobě konceptuálního základu pro konstrukci procesního modelu dotazování.

Brass a Goldberg (Brass & Goldberg, 2006) se zaměřili na kategorizaci sémantických chyb, jichž se dopouštějí studenti při formulaci dotazů. Sémanticky chybné dotazy definovali jako dotazy syntakticky správné, ale nevracející výsledek, jejž autor očekával. Sémanticky chybné dotazy mohou být například dotazy vracející prázdnou množinu. Systém řízení báze dat sémanticky chybné dotazy nerozlišuje a provádí je bez varování. Brass a Goldberg sestavili seznam podmínek indikujících sémantické chyby. V rámci svého výzkumu konstatují, že „nelze zjistit všechny sémanticky chybné dotazy, ale většina z nich by mohla být automaticky identifikována“. Autoři článku kromě výše uvedeného výzkumu provedli analýzu a návrh mechanismů, jež by mohly zjišťovat sémantickou správnost dotazů. Kromě toho se domnívají, že integrace sémantické kontroly do existujících systémů řízení báze dat by byla prospěšná. Přepokládaný výsledek výzkumu implementace sémantické kontroly dotazů byl především v oblasti snižování nákladů na vývoj programu účelným a smysluplným využitím databázového myšlení. Ve vztahu k této práci jsou závěry jejich zkoumání inspirující při stanovování zdroje sémantických chyb. Jejich rozdělení tak může být s ohledem na zdroj chyby na zbytečně komplikované, nedostatečně formulované, porušující standardní vzory, s přílišným zdvojením a chyby předčasně ukončující provádění dotazu. Výše uvedená klasifikace se tak stává integritním doplňkem kategorizace syntaktických chyb, jež byla použita v oblasti analytického zpracování problémových oblastí rozvoje databázového myšlení.

Brooks (Brooks, 1980) se zabýval metodologií výzkumu chování programátora. Ve svém článku popisuje aplikaci psychologických postupů při analýze pracovních postupů souvisejících s programovacími technikami. Došel k názoru, že hlavní faktor úspěchu ve výuce programování je právě efektivní použití správné metodologie. Zaměřil se tedy na rozbor vztahu předmětu výuky, materiálů použitých ve výuce a technik měření výsledků učení.

V rámci výzkumu programovacích konstruktů a technik existujících studií, které přímo sledovaly a manipulovaly chování programátora, zkoumaly program nebo společenskou dynamiku skupiny programátorů. Podle mého názoru je však hlavní přínos Brooksovy práce v oblasti konstrukce, výběru a použití programovacích nástrojů. Jejich rozbor založil na analýze modelů chování. Ty se tak staly základem při implementaci nových funkcí programovacího jazyka a při návrhu metod pro generování nových programovacích nástrojů nebo technik. Ohnisko jeho zájmu však zahrnovalo i oblast měřitelných vlastností programu. Ty jsou důležité při stanovení výkonnostních výsledků učení studenta pomocí měření. Kvalitu programu navrhoval měřit nejenom dobou vykonávání, ale i srozumitelností nebo dobou potřebnou k jeho odladění a změně.

Kromě toho přišel s teorií hierarchické struktury modelu učení programování. U toho určil zvyšující se komplexitu odspoda nahoru. Tu popsal jako znalostní strukturu vertikálně měnící své vnitřní uspořádání do stále složitějších a větších jednotek. Domníval se, že učení programovacího jazyka lze popsat jako proces založený na potřebě zvládnout dostatečné množství informací pro rekonstrukci programu. Při tom se domníval, že se jedná nejenom o znalost nízkoúrovňových detailů, jako jsou jména proměnných, konstant a výrazů či vysokoúrovňových principů, algoritmů a struktur. Celkové porozumění programu spatřoval v pochopení informací vyšší úrovně programovacích konceptů. Výše uvedené principy jsem aktivně využil při konstrukci obecného modelu rozvoje databázového myšlení. Konkrétně při stanovení struktury hierarchie odborných pojmů.

4.2 Teoretická východiska výuky databázového myšlení

Základním předpokladem pro výzkum učení dotazovacího jazyka je znalost jeho struktury, nad kterou lze formulovat princip fungování kognitivních operací a plánovat jejich rozvoj. Struktury programovacích jazyků se liší ve způsobu poznání, používání a odlišnými principy. Výzkum struktury dotazovacího jazyka je úzce spojen se zkoumáním principů učení jazyka obecně. Již v první polovině osmdesátých let Smith (Smith & Sage, 1983) hledal odpověď na otázku ohledně změny produktu společnosti z výroby hmotných statků na zpracování

informací. Pokoušel se určit, jaký má změna dopad na obsah informační gramotnosti. Předpověděl, že v budoucnosti se vzdělávání posune od rigidního učení celoživotně platných znalostí a dovedností k učení zaměřenému na flexibilní přizpůsobení neurčitosti. Důsledek posunu v neodvratné nezbytnosti zvyšování úrovně informační gramotnosti vyvolává otázky týkající se jejich naplnění. Na ty hledala odpověď řada vědců. Například Mayer ve své studii z roku 1979 (Mayer, 1979) rozlišuje úrovně poznání programovacího jazyka.

Později Mayer (Mayer, 1981) určil základní rámec příjmu technické informace. Vytvořil model kognitivního systému člověka popisující spolupráci krátkodobé a dlouhodobé paměti. Přijetí nové technické informace je při vstupu do lidského kognitivního systému postupně vstřebávána fázemi vnímání, dostupnosti a aktivace. Při vnímání musí žák přicházející informaci věnovat pozornost, aby si ji byl schopen uložit do krátkodobé paměti. Po jejím zapamatování v krátkodobé paměti vyvolá asociovanou teoretickou znalost z dlouhodobé paměti a tu použije k asimilaci nové informace, čímž se stane dostupnou pro další využití. Aktivace probíhá tak, že student pružně spojí novou teoretickou znalost s již existující teoretickou znalostí. Popsaný způsob procesu příjmu informace se nazývá smysluplné učení. Mayer navrhl model vlivu znalosti schématu počítače na učení. Následně provedl jeho experimentální ověření. Zjistil souvislost mezi dodržením kroků smysluplného učení a spojení nové informace s již existující znalostní strukturou. Při nedodržení smysluplného učení si žák novou informaci zapamatuje jako oddělenou položku v paměti.

V roce 1988 se Mayer (Mayer & Bayman, 1988) vrátil k problematice učení programovacího jazyka a rozpracoval otázky jeho vlivu na zlepšení syntaktické, konceptuální a strategické znalosti. Prováděl pokusy, ve kterých manipuloval s příručkami použitými ve výuce, a zjistil souvislost mezi kvantitou sémantické informace a chybováním studentů. Čím větší objem sémantické informace studenti měli k dispozici, tím méně dělali chyb. Kromě toho se zlepšovala jejich schopnost řešit obecné úlohy programování. V obecné rovině se kognitivním modelem příjmu informací zabýval Biffah (Biffah et al., 1997). Zkoumal způsob vnitřní reprezentace datových struktur v mysli studentů.

Kategorizaci neprocedurálních programovacích jazyků z hlediska jejich učení sestavil Ogden (Ogden & Brooks, 1983). Neprocedurální programovací jazyky rozdělil jednak na základě jejich syntaktické struktury a za druhé podle počtu výrazů. Zaměřil se především na rozdíl mezi formálním dotazovacím a přirozeným jazykem. Formuloval vztah mezi studiem formálního jazyka a rozvojem schopnosti argumentace. Vyjadřování ve formálním jazyce rozvíjí schopnost precizního a jasného způsobu komunikace. Domnívá se, že pokud by byla komunikace s počítačem přiblížena vyjadřování v přirozeném jazyce, dovolilo by to

komunikaci s počítačem většímu okruhu lidí. Ogden definoval podmínky pro osvojení znalosti formálního jazyka. Podle něj je důležitý explicitní model databáze, databázové atributy a způsob propojení atributů navzájem.

Podle Ogdena k dosažení úspěšnosti v 75 % případech je k osvojení znalosti formálního jazyka zapotřebí 10 až 20 hodin tréninku. Praktická zkušenost odstranila překlepy, chyby v synonymech, syntaktické chyby, chyby v tečkách a chyby v jazyku. Dotazy ve formálních jazycích byly často příliš podrobné a komplikované. V oblasti komparace přirozeného a formálního jazyka vzniklo mnoho laboratorních studií prototypů přirozeného dotazovacího jazyka, oborové studie prototypových systémů a popisy komerčně dostupných produktů pro přirozený jazyk. Výzkum pohledu zaměřeného na náklady osvojení znalosti programovacího jazyka zkoumal Turner (Turner et al., 1985). Ve své práci kategorizoval uživatelské rozhraní dotazovacího jazyka podle nároků uživatele. Vhodnost dotazovacích jazyků srovnával podle poměru ceny a výkonu. Cenu určil jako časovou náročnost a vstupní motivaci uživatele při zaškolení. Náklady vyjádřil jako míru vůle rozvíjet schopnosti v daném jazyce. Hlavní oblasti výhod dotazovacího jazyka spatřuje Turner v oblastech kompozice výsledků, oblasti prezentace výstupů a oblasti flexibility interakce.

Z toho vyplývají kritéria pro vyhodnocování použitelnosti programovacího jazyka. Tu lze hodnotit především mírou náročnosti na naučení jazyka, náročností jeho používání a shodou programovacího jazyka s typem úloh, které jsou jím řešeny. Soloway (Soloway, 1986) se v souvislosti s programováním zmiňuje o skryté znalosti, kterou programátoři označují jako intuici, pocit apod. Za jádro programování považuje především učení myšlenkových konstrukcí struktur a učení schopnosti vysvětlovat principy a mechanismy. Na základě poznání struktury, kategorizace a ceny znalosti dotazovacích a programovacích jazyků se vědci pokoušeli hledat vhodné metody jejich výuky.

Jedním z průkopníků na tomto poli byl Seymour Papert. Ten se ve své knize zabýval přiblížení počítačů dětem. Položil tím základy pro návrh a implementaci programovacích jazyků určených hlavně pro primární výuku programování (Papert, 1980). Myšlenka propedeutického programovacího jazyka je založena na schopnosti žáka rozvíjet aktivní řešení problémových úloh. Programovací jazyk LOGO poskytl prostředí pro rozvoj kompetencí potřebných k úspěšnému řešení problémových, zejména matematických, úloh a programování. Implementace byla založena na vytvoření umělého prostředí, ve kterém existuje manipulovatelný objekt. Ten bylo možné řídit příkazy. Zároveň bylo možné skládáním dílčích příkazů vytvářet příkazy nové. Mezi jeho přímé následníky patřil například programovací jazyk Turtlegons (Culberson & Rawlins, 1985) nebo Sinclair LOGO (Sparer,

1984). V související studii se Layman (Layman & Hall, 1988) zabývá problematikou programovacího stylu osvojeného programátory, jejichž první programovací jazyk bylo LOGO.

4.3 Aplikační domény kognitivního modelu rozvoje databázového myšlení

Za stěžejní aplikační doménu kognitivního modelu rozvoje databázového myšlení pro svou disertační práci považuji systematizaci databázového myšlení. Tu provádím na základě analýzy kognitivních modelů procesu dotazování. Výsledkem je potom pohled na databázové myšlení umožňující vyvinutí teoretického konceptu aktivit sloužících k rozvoji databázového myšlení. Jako dílčí aplikační doména rozpracovaná za účelem upřesnění modelu dotazování je rozbor chybování při učení dotazovacího jazyka. Model chybování je obecně uplatnitelný a může být uplatněn například při tvorbě inteligentního vyučovacího systému.

Při hledání kognitivního modelu chyb dotazování je klíčovou komponentou hierarchie chyb, podle které se program rozhoduje o dalších krocích. Autory, kteří se zabývají tématem chybování, jsou například Weerasinghe (Weerasinghe et al., 2008) a Wu (Wu, 2008). Na základě analýzy koncipoval Wu analytickou jednotku. Jeho názor je, že proces zotavení z chyb začíná ve chvíli, kdy uživatel zjistí, že chyboval, a provede korekci zjištěné chyby. Každá oprava chyby je pokus o opravení chyby a může se skládat z jedné nebo série akcí. Chyba a následující pokusy o opravení chyby jsou definovány jako chybová epizoda (viz obrázek (Wu, 2008), str. 94). Trvání chybové epizody začíná v bodě, kdy uživatel zjistil chybu, a končí v bodě, kdy uživatel uspěje nebo vzdá opravu. Každá chybová epizoda zahrnuje jeden nebo více pokusů o opravu. Každý pokus o opravu může být neúspěšný a uživatel může pokračovat dalším pokusem o opravu. Neúspěšný pokus o opravu může vyústit v chybovou hlášku. Chybová epizoda může být kvantifikována například trváním a celkovým počtem pokusů o opravu. Pokus o opravu je kvantifikován intervalem mezi dvěma systémovými hláškami.

Kromě toho Wu (Wu, 2008) stanovil taxonomii chyb vznikajících při učení SQL na:

1) „Slips and lapses“ – účastníci vykonali správnou myšlenku chybně. Například účastník zadal dva středníky na konci SQL skriptu,
2) Syntaktické chyby jako příležitosti, ve kterých účastník použil nesprávně syntaktické pravidlo. Může to být nesprávné zařazení uspořádání před omezující podmínku,
3) Sémantické chyby – chyby, jejichž příčinou byla neúplná nebo nedostatečná znalost nebo omezené zdroje. Chyba tohoto typu je například left join místo inner join.

Při měření rychlosti zotavení z chyb byly vyhodnocovány dvě hlavní dimenze:

1) celkový čas strávený v chybové epizodě,
2) celkový počet chyb v rámci epizody.

Při analýze chyb Wu (Wu, 2008) identifikoval namáhavé nebo složité problémy, které se vyznačovaly buď tím, že s nimi uživatelé strávili příliš mnoho času, nebo provedli příliš mnoho neúspěšných pokusů.

Druhou aplikační doménou kognitivních modelů rozvoje databázového myšlení v souvislosti s rozvojem obsahu výuky je oblast prostředků pro podporu výuku. Přesvědčení, ze kterého pramení motivace k aplikaci přístupu rozvoje prostředků pro podporu výuky, je především stanovisko kladoucí důraz na určení směru rozvoje kompetencí budoucích učitelů. Stejný názor uvádí ve své práci Jukic (Jukic & Gray, 2008). Podle něj by se výuka budoucích učitelů měla zaměřit především na průpravu v oblasti rozvoje praktických dovedností nezbytných ke zvládnutí výuky. Těm by se podle jeho názoru mělo dát přednost před rozvojem dovedností souvisejících s praktickou aplikací poznatků do vytváření programů. Směr soustředěný na cílené zkoumání principů databázového myšlení je základním konceptem pro zakotvení dalšího rozvoje vzdělávací soustavy. Na jedné straně existuje přístup kladoucí důraz na zvyšování statusu a rozvoj obecných dovedností pedagogických odborníků. (Neumajer, 2007) (Zákon číslo 561/2004 Sb., o předškolním, základním, středním, vyšším odborném a jiném vzdělávání (školský zákon)).

Opačný názorový proud ke zvyšování odbornosti učitelů spíše než k jejich pedagogické přípravě zastává Murray (Murray & Guimaraes, 2008). Navrhuje rozšířit standardní obsah kurikula kurzu databázových technologií. Považuje za žádoucí zařadit do něj pokročilá témata, jako jsou datové sklady, datová výměna přes XML, databázová bezpečnost, architektura databázových systémů a ladění výkonu. Pro rozšíření obsahu předmětu databázových systémů argumentuje nutností zahrnutí výše uvedených témat do osnov. Vzdělávání rozvíjející učitelovu specializaci v oblasti databázových systémů dle jeho názoru však není pro učitele příliš atraktivní (viz například (Neumajer, 2007)).

Dle mého názoru je však rozvoj databázového myšlení důležitým tématem didaktiky informační výchovy. Považuji jej za podstatný a klíčový, neboť tvoří kostru informační gramotnosti dnešního člověka. Jako příklad lze uvést jejich prezenci v ECDL, EPICT, ETS, ICT kompetence ve státech EU atd. V rámci ECDL je to modul databáze, který je zaměřený

na filozofii práce s databázemi, vytvoření seznamu, definice textových a číselných polí, výběr položek a jejich zpracování, ukládání dat. EPICT na databáze pohlíží jako na nástroj, jenž umožňuje vyhledávání na internetu, hledání titulů v knihovním systému nebo rezervaci letenek. Aplikace osvojení databázového myšlení tudíž není omezena pouze na procházení webových stránek a zadávání klíčových slov do vyhledávacích služeb. Databázové myšlení lze tedy aplikovat do mnoha oborů lidské činnosti. Pro ETS rozvoj databázového myšlení je především zvládnutí vyhledávání informací prostřednictvím rozhraní databázového systému. V rámci ICT kompetence učitelů ve státech EU se jedná o jednu z klíčových ICT dovedností spadající do kategorie použití kancelářského softwaru. Další standard je NEA Technology Taxonomy, ve které lze rozvoj databázového myšlení nalézt ve třetí úrovni. Ta obsahuje kompetence pro používání souboru softwarových nástrojů, tzn. tabulek, databází, prezentací apod. V neposlední řadě se rozvojem databázového myšlení zabýval SIPVZ, ve kterém byl samostatný vzdělávací modul nazvaný databázové systémy (modul P v roce 2005). Zahrnoval zpracování dat v podobě jejich shromažďování, zpracování a interpretaci. Na základě výše uvedených příkladů konstatuji, že zvládnutí databázového myšlení patří do profesního standardu učitele. Mám tím na mysli pojetí, jak jej chápe Rýdl (Rýdl et al., 2009) (standard kvality učitele jako rámec žádoucích kompetencí a činností v navrhovaných ukazatelích).

Při zařazování databázového myšlení do portfolia kompetencí učitelů ICT je nutné prozkoumat přilehlé okolí vyučování nejenom dotazovacího jazyka, ale i strukturovaného programování. Seyed-Abbassi (Seyed-Abbassi, 1993) zkoumal vliv použití projektu ve strukturovaném dotazovacím jazyce v rámci kurzu informačních systémů. Dospěl k názoru, že projekt rozšiřuje studentovo porozumění teoretickým konceptům a povědomí studentů o náročnosti a složitosti použití počítačových technologií v praxi. Tento poznatek později aplikoval do aktivity pro hledání odpovědi na otázku, jakým způsobem lze databázové myšlení implementovat do výuky.

Jedním z průkopníků praktického aspektu vyučování byl v první polovině osmdesátých let Citron (Citron, 1983). Popsal trend změny kurzů počítačových věd vstříc k jejich praktičnosti. Vyslovil nutnost reagovat na požadavky zaměstnavatelů. Ti v té době nepreferovali teoreticky orientované zaměstnance. Citron zastává názor, že právě porozumění kontextu problému je klíčem k jeho vyřešení. Kushan (Kushan, 1994) se věnoval přípravě budoucích učitelů programování. Podal přehled o pokusech definice kvality instruktorů programování různých expertních skupin, které se pokusily ustanovit, jaké znalosti by měli mít budoucí učitelé programování. Profesionální vědci sestavili několik pracovních skupin, které doporučily programy ke studiu. Několik států USA vyvinulo standardy pro certifikáty výuky

počítačových kurzů a univerzity navrhly programy studie pro přípravu učitelů informatiky. Tyto programy se však od sebe navzájem velice lišily.

Kushan (Kushan, 1994) uvádí kompetence, které by měl budoucí učitel programování znát. Základem byla trojice nepostradatelných znalostí pro budoucího učitele programování. Byly to kompetence k řešení problémů, pedagogické znalosti a programátorské znalosti. Mezi koncepty, které by měly být zahrnuty do vyučovacích hodin – to je znalost syntaxe programovacího jazyka, sémantická znalost, přehled o strukturovaném programování, top-down návrhu, modularity, dokumentování, návrhových nástrojů, techniky odlaďování a simulace testování. Dále Kushan sestavil stručný přehled změn v rozvoji přístupu učitelů k definici obsahu výuky. Klasifikoval je do tří vývojových fází. V první fázi se na středních školách vyučovala především syntaxe. Hlavní důvod byl, že většina učitelů se rekrutovala z řad bývalých programátorů. Ti vyučovali především psaní programů. Důležité bylo, aby program fungoval a produkoval správný výstup. Takový typ výuky byl na úkor znalosti technik řešení problému. Kushan (Kushan, 1994) se pokusil formulovat výsledky výuky programování. Zdůrazňoval důležitost přínosu v podobě procvičování strategií řešení problému, poznatků o vlastním myšlení, učení matematiky (především geometrie), poznatky o sociální interakci a učení o hodnotách. Podle Callistera (Callister & Burbules, 1990) je nezbytné kurikulum studentů učitelství ICT revidovat z pohledu jejich budoucího uplatnění. Navrhuje přesun pozornosti od výuky obecných kompetencí ke konkrétním aplikačním doménám. V rámci výuky obecných předmětů navrhuje zvýšit zapojení počítačů do výuky. Zároveň doporučuje u studentů učitelství ICT položit důraz na otázky efektu používání počítačů ze sociologického i psychologického hlediska jejich vlivu na učení.

Otázkou přípravy na používání počítačů se zabývá Handler (Handler, 1993). Ve svém článku si položil otázku, jakým způsobem je nezbytné definovat přípravu učitelů. Otázka obsahu učiva studentů učitelství ICT v sobě zahrnuje vymezení počítačové gramotnosti a navržení funkčního modelu, podle kterého by bylo možné definovat kurikulum. V souvislosti s vlivem nástupu mikropočítačů se Hazari (Hazari, 1991) pokusil navrhnout funkční model výuky informační gramotnosti a své závěry ověřil experimentem. Přínos výuky a výslednou strukturou znalosti v paměti žáka se zabýval Gagne (Gagne & White, 1978). Podle jeho názoru jsou výsledné znalosti v úzké souvislosti s dovedností žáka programovat. Matematické schopnosti jsou podle jeho názoru premisou žákovy schopnosti učit se nová logická pravidla. Žákovy jazykové schopnosti jsou oproti tomu úzce spojeny se schopností učit se syntaktická pravidla (Sauter, 1986).

Bottino (Bottino, 1992) se věnoval pedagogickému hledisku různých přístupů

k programování. Zkoumal rozdíl přínosu učení u dvou různých programovacích jazyků. Thornburg (Thornburg, 1988) si myslel, že základní rozdíl mezi vysokoúrovňovými programovacími jazyky nespočívá v rozdílné syntaxi, gramatice nebo slovnících, ale v odlišných metaforách. Například žák studující procedurální jazyky si představuje program jako soubor procedur, ale pokud je žákovi vysvětlován objektový jazyk, program je pro něj reprezentován objekty a zprávami.

Vývojem diagnostických testů zkoumajících úroveň počítačové gramotnosti se ve svém článku zabýval již Pyrczak (Pyrczak, 1990). Informační gramotnost vymezil jako:

1) používání aplikačního programu,
2) sociální kontext používání počítače,
3) základní porozumění, co je počítač a jak funguje,
4) znalosti v oblasti historie počítačů,
5) zkušenost s alespoň jedním vysokoúrovňovým programovacím jazykem.

K definici informační gramotnosti existuje mnoho různých přístupů a názorových proudů. Obecně lze konstatovat, že každý autor ji definuje jinak. Jedním z průkopníků na tomto poli byl Kurtz (Kurtz & Adams, 1988). Ten za stěžejní úlohu úvodního kurzu počítačových věd považoval oblast zdokonalování kompetencí pro řešení problému a programování. Oproti tomu programování nepovažoval za nejdůležitější složku obsahu výuky. Zabýval se problematikou principu expanze (expanduje myšlenku z její jednoduché podoby do složitějších podob) a integrace úrovně spojující několik konceptů na stejné úrovni obtížnosti. Rozvoj těchto konceptů konkretizoval na daných příkladech v oblasti kontroly toku, rekurze a skrývání informací. Výsledek výzkumu přispěl ke stanovení analytického cíle vedoucímu k nalezení rovnováhy mezi schopnostmi řešit problém a odbornými schopnostmi v oblasti rozvoje databázového myšlení.

Obsahově orientovanou práci publikoval i Sauter (Sauter, 1986). Ten se zabýval predikcí znalostí, které jsou pro rozvoj databázového myšlení klíčové. V době, kdy psal svůj článek, byl hlavní oporou paralely rozvoje databázového myšlení s rozvojem matematických a jazykových dovedností. Domníval se, že matematické schopnosti mají úzkou vazbu na schopnost učit se logická pravidla. Oproti tomu jazykové znalosti spojoval se schopností učit se syntaktická pravidla. Významný vliv na směr vývoje moderních programovacích jazyků a jejich výuku měl Papert. Ten v oblasti programovacích jazyků publikoval přelomovou práci Mindstorms (Papert, 1980). Zabýval se v ní rozpracováním myšlenky, kam má směřovat

výuka programování u žáků. Byl toho názoru, že rozvoj procedurálního přístupu je pro žáky důležitou možností poznat své vlastní myšlení. Stanovil tím jasný vzdělávací cíl. Ten později vedl k instrukcionálním metodám výuky a zároveň poskytl prostor pro rozvoj myšlení v nestrukturovaném prostředí. Další výzkumy však ukázaly, že výukové prostředí jazyka LOGO může být pro některé žáky příliš složité a přínos při přenosu nabytých znalostí do jiných oblastí učení je marginální. Na výuku programovacího jazyka LOGO navázala etapa výuky programování, s důrazem na metody vedení výuky učitelem. Byl tím položen základ hlubšího rozvoje etablujících se metod výuky. Zároveň to vedlo ke zpomalení tempa objevování nových metod výuky. Učitelé se domnívali, že pokud mají studenty naučit zvládnout technickou stánku rozvoje databázového myšlení, musí být do výuky zahrnuto její praktické procvičování. Pokud bude taková výuka vyučována přímo učitelem, dojde tak k přenosu dovedností řešit problém.

Tím byly položeny základy preference strukturovaného programování ve výuce. Fisher (Fisher, 2008) zpětně prováděl evaluaci motivace vedoucí k použití různých programovacích jazyků ve výuce. Dospěl k názoru, že jedním z faktorů pro efektivní rozvoj programátorského myšlení je schopnost stanovit relevanci programovacího jazyka v kontextu konkrétní situace. Kromě toho nalezl pro každý programovací jazyk unikátní vlastnosti, které jej determinovaly pro řešení určitého specifického okruhu problémů. Z toho vyvodil ukazatel na zdroj odlišných koncepcí jednotlivých programovacích jazyků. Ten je tvořen různým teoretickým základem. Charakteristickou determinantou původu dotazovacího jazyka je dle jeho názoru relační algebra. Souhrnem výše uvedených poznatků z teorie taxonomie programovacích jazyků dospěl k závěru, že v rámci rozvoje programování by mělo být vyučováno více jazyků odlišných po skladební i obsahové stránce. Hlavním účelem jejich výuky je potom rozvoj schopnosti žáka zvládnout kompetence cílené na efektivní učení nových počítačových jazyků. Prakticky je výše uvedený efekt založený na principu snadnějšího osvojení nového počítačového jazyka v případě, kdy student již některý programovací jazyk ovládá. Rozvoj společné kompetenční báze více než jednoho programovacího jazyka umožňuje schopnost určit a evaluovat silné a slabé stránky jednotlivých jazyků. Využitím této znalosti může student vyhodnotit problém a podle něj vybrat vhodný programovací jazyk.

Aby byl umožněn výzkum učení dotazovacího jazyka, je účelné rozpracovat jeho znalostní strukturu. Ta může být základem pro formulaci principů fungování kognitivních operací, na jejichž základě jsou potom stanovovány aktivity vedoucí k rozvoji databázového myšlení. Struktury jazyků jsou odlišné v efektivních metodách vedoucích k jejich poznání, použití a principům. Výzkum struktury dotazovacího jazyka je úzce svázán s poznáním principů

metod učení jazyka. Hledání významu informační gramotnost vyvolává otázky o jejím naplnění. Odpověď na ně i v minulosti hledala řada vědců.

Mezi ně patří například Mayerův názor, jejž publikoval ve své studii v roce 1979 (Mayer, 1979). Stanovil v ní úrovně poznání programovacího jazyka. Později Mayer (Mayer, 1981) svůj koncept rozšířil a určil základní rámec příjmu technické informace. Kromě toho vyvinul model kognitivního systému člověka pracující s konceptem kooperace krátkodobé a dlouhodobé paměti. Nové technické informace do lidského kognitivního systému podle jeho názoru procházejí fázemi vnímání, dostupnosti a aktivace. Při vnímání musí student věnovat pozornost přicházející informaci. Potom následuje ukládání do krátkodobé paměti. Po uložení informace do krátkodobé paměti dojde k vyvolání související teoretické znalosti z dlouhodobé paměti. Ta je později použita k asimilaci nové informace. Při aktivaci poznatků musí student pružně spojit novou teoretickou znalost s již existující znalostní strukturou. Fáze, ve kterých dochází k integrování nových informací, jsou nazývány smysluplným učením. Mayer teorii experimentálně ověřil studií vlivu znalosti schématu počítače v souvislosti s technickými předpoklady pro učení. Podle jeho názoru dochází v momentě, kdy je libovolný z kroků smysluplného učení vynechán, k zapamatování nové informace jako oddělené položky v paměti.

V roce 1988 se Mayer (Mayer & Bayman, 1988) vrátil k problematice rozvoje znalosti programovacího jazyka a rozpracoval otázky vlivu obsahu výuky na zlepšení syntaktické, konceptuální a strategické znalosti. Experimentoval s modifikací obsahu výukových materiálů ve výuce. Zjistil souvislost mezi množstvím sémantické informace předkládané žákovi a jeho chybováním v kontrolním testu. Čím více sémantických informací studenti měli k dispozici, tím méně dělali chyb. Kromě toho kvantita sémanticky orientovaných podkladů obsažených ve výukových materiálech měla pozitivní vliv na žákovu schopnost řešit obecné programovací úlohy.

V obecné rovině kognitivní model příjmu informací studoval Biffah (Biffah et al., 1997). Analyzoval vnitřní uspořádání reprezentace datových struktur v mysli žáka. Kategorizaci neprocedurálních jazyků v rámci výzkumu učení dotazovacího jazyka sestavil Ogden (Ogden & Brooks, 1983). Rozlišoval dva základní atributy relevantní pro rozvoj poznatků v oblasti neprocedurálních jazyků. První z nich byla syntaktická struktura a druhý byl počet výrazů. Upozornil na rozdíl mezi formálním dotazovacím a přirozeným jazykem. Domníval se, že studium formálního jazyka rozvíjí argumentační schopnosti jedince. Vyjadřování ve formálním jazyce vede k rozvoji precizní a jasné komunikace. Oproti tomu komunikace s počítačem v přirozeném jazyce dovoluje, aby se komunikace účastnil větší okruh lidí.

Ogden tak sestavil bázi podmínek, jež dovolují osvojit si formální jazyk. Z jeho pohledu je důležitý explicitní model databáze, databázové atributy a způsob propojení atributů navzájem. Ogden se domníval, že k dosažení úspěšnosti v 75 % případů je k osvojení znalosti formálního jazyka zapotřebí 10 až 20 hodin tréninku. Praktická zkušenost odstranila překlepy, chyby v synonymech, syntaktické chyby, chyby v tečkách a chyby v jazyku. Dotazy ve formálních jazycích mohly být často příliš podrobné a komplikované. V rámci srovnání přirozeného a formální jazyka existovaly laboratorní studie prototypů přirozeného dotazovacího jazyka, oborové studie prototypových systémů a popisy komerčně dostupných produktů pro přirozený jazyk.

Evaluaci nákladové zátěže pro osvojení znalosti programovacího jazyka provedl Turner (Turner et al., 1985). Ten uživatelské rozhraní dotazovacího jazyka položil do úzké souvislosti s nároky uživatele. Aby byl schopen srovnávat dotazovací jazyky, stanovil si míru vyjadřující poměr ceny a výkonu. Cenu vyjádřil mírou vstupní motivace uživatele při zaškolení. Náklady vyjádřil jako míru vůle rozvíjet schopnosti v daném jazyce. Hlavní výhody jazyka spatřuje Turner v možnosti kompozice výsledků, prezentace výstupu a flexibilitě interakce. Z toho vyplývají i kritéria pro hodnocení použitelnosti programovacího jazyka. Ta se týká náročnosti naučení jazyka, jeho používání a shody programovacího jazyka s typem úloh, které jsou jím řešeny.

Další oblastí s úzkou vazbou na formulaci obsahu výuky databázového myšlení je otázka obsahu kurzu programování pro učitele. George (George & Valeva, 2006) považuje za komplementární součást obsahu výuky databázového kurzu kromě základních principů databázových systémů i otázky bezpečnosti. Udoh (Udoh, 2006) zastává názor, že databázový kurz by se měl skládat ze dvou částí. První část by měla zahrnovat teoretické základy databázových systémů. Do nich spadá teorie relačního modelu, entit relačního modelování, normalizace, dotazovacího jazyka, relační algebry a počtu, organizace úložiště a optimalizace dotazů. V pokročilém kurzu potom navrhuje vyučovat transakce, kontrolu konkurence, zotavení, datové sklady, bezpečnost, distribuované a objektové databáze, data mining a administraci databázových systémů. Dekeyser (Dekeyser et al., 2007) se domnívá, že cílem koncepce obsahu výuky databázového myšlení není pouze vybudovat schopnost psát gramaticky správné dotazy, ale i překládat otázky z přirozeného jazyka do výrazů v dotazovacím jazyce.

Meeneker (Meeker & Nohl, 2007) si povšiml problému, jenž se vyskytoval ve výuce databázového myšlení u studentů bakalářského studia. Tím byl nedostatek času na podrobné vysvětlení kladených otázek v rámci praktických cvičení rozvoje databázového myšlení. Při

hledání možných způsobů řešení identifikoval dvě hlavní varianty. První varianta byla přidat do kurzů laboratorní cvičení. Druhá varianta byla implementace mini-projektu do osnov kurzu. V souvislosti s tím se Cook (Cook, 2008) zmiňuje o obtížích vyplývajících z problémů při sestavení efektivního kurikula. Při výzkumu problematických oblastí v koncepci obsahu učiva pro rozvoj databázového myšlení narazil na problém, jejž označil pojmem indoktrinace. Tím je vliv nástrojů, které používáme, na nezaujatý přístup k řešení problémů. Prakticky lze jev sledovat u žáka, který se pokouší aplikovat korektní pracovní postup osvojený z některého naučeného programovacího jazyka na nově naučený programovací jazyk. To může být problém, pokud se paradigma dříve naučeného programovacího jazyka odlišuje od paradigmatu nově naučeného programovacího jazyka. Další problém pojmenoval souvislost kvantity a efekt měřítka. Efekt se vyskytne, pokud student použije nabyté znalosti pro vývoj rozsáhlých a objemných informačních systémů a při něm dojde k výskytu chyb, ke kterým by při vývoji u menšího systému nedošlo. Indoktrinace je základním konceptem, na který je nutné brát ohled při rozvoji databázového myšlení. Její častý výskyt vyplývá z volby pořadí výuky programovacích jazyků, kde je jako první většinou vyučován procedurální programovací jazyk a teprve potom dochází k primárnímu rozvoji databázového myšlení. V tom případě je reálná šance, že žák bude aplikovat poznatky o procedurálním jazyce i do učení neprocedurálního dotazovacího jazyka.

4.4 Didaktické prostředky výuky databázového myšlení

Jedním z moderních didaktických prostředků určených k rozvoji databázového systému vyvinul Guimaraes (Guimaraes & Murray, 2007). Stal se spoluautorem balíku animovaných kurzů databázové bezpečnosti, který obsahuje přes 70 výukových programů zahrnujících výukové animace, didaktické texty a praktická cvičení. Domnívá se, že použitím názorných animací může být proveden efektivnější a názornější výklad témat, která jsou představována ve třídě. Zároveň kurz poskytuje místo pro procvičování a sběr zpětné vazby. Guimaraes (Guimaraes, 2006) se v minulosti zabýval prostředkem pro podporu výuky databází pod názvem Kennesaw Database Courseware (KDC). Jedná se o počítačový kurz vytvořený se záměrem podpořit výuku použitím příkladných počítačových animací. Kurz se skládá ze tří hlavních modulů (návrh databáze, SQL dotazy a transakční zpracování). Dekeyser (Dekeyser et al., 2007) se domnívá, že zařazení praktické práce s databází do výuky má výhodu především v podobě okamžité zpětné vazby na syntaktickou správnost zadaného výrazu. Výuka obsahující pouze zpětnou vazbu na syntaktickou správnost pokusů studenta však není dostačující pro přípravu a prověřování znalostí, neboť v tom případě je důležité položit důraz

i na sémantickou správnost zadávaných dotazů. Dále se ve své práci věnuje porovnávání různých nástrojů pro výuku dotazovacího jazyka. Ze zkušeností, které načerpal při srovnávání výukových nástrojů, vyšel při stanovování požadavků na nový nástroj určený k výuce databází. Požadavky definoval následovně:

1) Bohatá, automatická zpětná vazba,
2) prezentace relačního schématu za účelem snížení paměťových nároků kladených na žáka,
3) podrobný rozbor provádění dotazu, aby bylo dosaženo hlubokého porozumění fungování systému,
4) hodnocení ostatních žáků za účelem získání další zpětné vazby,
5) sestavení algoritmu, který by umožnil testovat dotazy studentů,
6) automatické vyhodnocení přesnosti vyjádřené počtem oprav provedených studenty,
7) nižší počet vstupů vyučujících při procvičování má za cíl snížit zátěž
8) zvýšení objektivity známek udělovaných studentům.

Brusilovsky (Brusilovsky et al., 2008) ve svém článku popisuje integraci různých nástrojů určených pro výuku dotazovacího jazyka do jednoho komplexního nástroje. Zastává názor, že pro důkladnou výuku dotazovacího jazyka nejsou dostačující pouze základní kompetence, ale je nezbytné procvičovat i další, hlubší a širší poznatky. Systémy pro podporu výuky dotazovacího jazyka rozdělil do dvou hlavních skupin. Do první skupiny zařadil programy názorně demonstrující základní principy SQL. Do druhé skupiny zahrnul systémy rozvíjející databázové myšlení řešením praktických problémů za použití dotazovacího jazyka. Při zkoumání možností integrace vyučovacích nástrojů si položil následující otázky: Jak ověřit totožnost uživatele vůči více různým systémům? Jak studentovy akce zaznamenat a spárovat je s aktivitami v dalších systémech? Jak může být u studenta zjištěna úroveň znalosti studenta spárování záznamů jeho aktivit v několika různých systémech? Systém, který vyvinul, nazval SQL Exploratorium. Skládal se ze tří modulů. První modul obsahoval komentované příklady, druhý testovací otázky a třetí SQL laboratoř. Když později provedl evaluaci zpětné vazby ohledně integračního systému, setkal se u studentů s pozitivním ohlasem.

McIntyre (McIntyre et al., 1995) se zabýval otázkou použití softwarových nástrojů ve výuce relačního databázového návrhu. Provedl experiment zaměřený na manipulaci se stupněm praktických rad ve výukových materiálech. Výsledný vliv předloženého studijního materiálu zkoumal na automatizovaném prostředku návrhu relační databáze. Překvapivě byly výsledky

v oblasti zájmu studentů, jejich zainteresovanosti a zvědavosti lepší, než očekával. Studenti byli studijními materiály podpořeni k lepším výkonům díky lepšímu pochopení a většímu úspěchu v praktických příkladech.

V oblasti prostředků pro výuku programování nelze nezmínit názor, který zveřejnil již Aston (Aston, 1988). Položil si otázku, zdali se v rámci mezinárodního programu vzdělávání mikroelektroniky dostatečně přispělo k rozšíření nových informačních technologií. Při hledání odpovědi dospěl k zajímavému názoru, že nestačí škole pouze dodat hardware, ale musí se třeba současně s ním dodat i software a doplňující vzdělávání. Fischer (Fischer et al., 1991) ve svém článku popsal kritický přístup ke konstrukci interaktivního znalostního systému. Kritické myšlení považoval za důležité při řešení problémů a učebních aktivitách. Položil tím teoretický základ znalostně orientovaných systémů a předurčil kontext pro toto paradigma. Boyd (Boyd, 1982) ve svém článku kategorizuje učení podporované počítačem do čtyř skupin a pokouší se odhadnout jejich možný dopad. Vývojové diagramy jsou spojeny s počátkem programování a v praxi přijímány. Mimo jiné jsou ale i časté terče útoků kritiky. Brooks (Brooks, 1980) tvrdí, že vývojové diagramy jsou špatné, jejich studium zabírá mnoho času a jedná se nejčastěji o předraženou část dokumentace programu. Mayerovi (Mayer, 1975) se v sérii experimentů nepodařilo zjistit, jaký mají vývojové diagramy vliv na sestavování, chápání, odlaďování a změny programů. Na základě negativních výsledků experimentu nedoporučuje vývojové diagramy používat jako podporu výuky programování. Mayer experimentálně zkoumal použití vývojových diagramů ve výuce začínajících programátorů. Ti, kteří neměli zkušenosti s programováním, byli podrobeni výuce zjednodušené verze programovacího jazyka. Někteří z nich používali stavové diagramy, jiní ne. Skupina, která měla k dispozici stavové diagramy, nebyla o mnoho lepší než skupina, která stavové diagramy nepoužívala. Dokonce se v některých úlohách projevila jako horší. Použitím entitně-relačního diagramu a jeho vztahu k anglickému jazyku se ve svém článku zabýval Wilson (Wilson, 1987). Popisuje problémy způsobené nasazením ER diagramu jako vhodného nástroje pro výuku databází. Definoval příčinnou souvislost mezi chybami v ER diagramu a kognitivními procesy při jejich konstrukci v anglickém jazyce. Rae (Rae, 1990) definoval základní kompetence nutné pro osvojení návrhu ER diagramů:

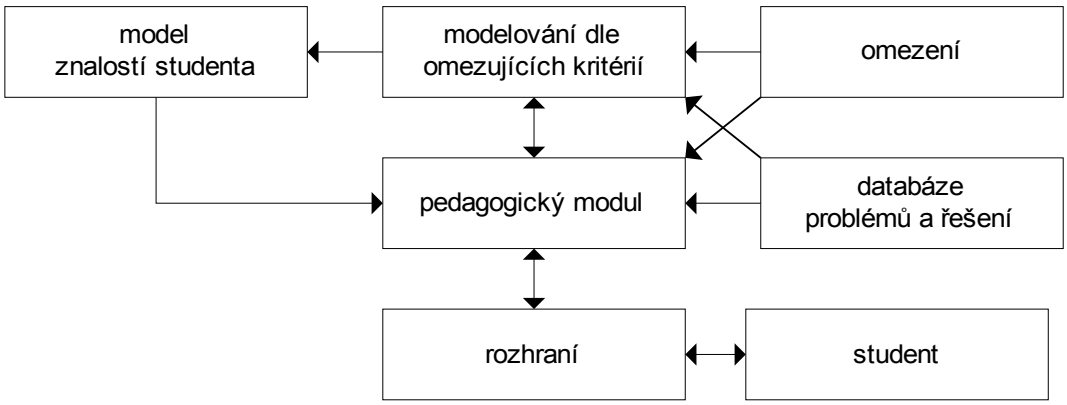
1) porozumět systému řízení báze dat (SŘDB),
2) ovládat základy relačního modelu,
3) naučit se popsat reálný svět ER modelem,
4) pochopit princip překladu ER modelu do dotazovacího jazyka.

S tvorbou vyučovacích systémů úzce souvisí problematika konstrukce kognitivního modelu studentů, podle kterého by mohla být řízena výuka. Self (Self, 1979) představuje některé otázky zaměřené na modelování znalostí studentů v rámci vyučované domény. Göktepe (Göktepe et al., 1989) se zabývá návrhem a implementací nástroje pro výuku programování. K tomu účelu provedl analýzu vyučovacích strategií, které definoval jako:

1) dril a praxe,
2) dialog,
3) simulace,
4) hry,
5) testování, řešení problému,
6) učení pomocí objevování.

Problematikou inteligentních vyučovacích systémů se zabýval Bowerman (Bowerman, 1992). Pojednává o systému pro výuku psaní v německém jazyce. Nástroj se chová stejně jako textový procesor, ale je navíc obohacen o možnost zjistit chybu a reagovat na ni vysvětlením. Boulet (Boulet, 1987) navrhoval vzdělávací software s diagnostickým přístupem. Kearns (Kearns et al., 1996) se věnoval problematice prostředků podporujících výuku dotazovacích jazyků. Zabýval se především systémem ESQL. Ten simuluje uživatelské rozhraní, ale nabízí detailnější zpětnou vazbu. Při chybě systém zobrazí instrukce a umožní korekci dotazu. Od jiných systémů se liší možností prohlédnout si posloupnost nízkoúrovňových operací, které vykonává databázovým systém. Tím je studentovi umožněno vyvinout lepší kognitivní model principu fungování databázového systému. Jiný systém sloužící k výuce databází popisuje Mitrovic (Mitrovic, 1998). Na University of Cantenbury byla v rámci Intelligent Tutoring Group (ITG) u zrodu hned několika inteligentních vyučovacích systémů Intelligent Tutoring System (ITS). Do skupiny vyučovacích systémů patří například SQL-Tutor. Je to systém pro výuku strukturovaného dotazovacího jazyka. Může být použit ve výuce rozvoje schopností řešení problému, konceptuálního učení a meta učení. Systém interně udržuje model studenta, podle kterého generuje pedagogické akce, a tím nabízí individuální přístup ke každému studentovi.

Jiný systém ITG, nazvaný KERMIT, podporuje výuku datového modelování. Výzkumnou zprávu o jeho fungování publikovala Suraweera (Suraweera & Mitrovic, 2002). Popisuje funkcionalitu systému KERMIT založenou na předchozích zkušenostech z ITG. Ta využívá poznatky ze systémů SQL-Tutor a CAPIT (systém pro výuku interpunkce). Ve zprávě se autorka zabývá především efektivitou nasazení systému do výuky. Další člen skupiny ITG Brent (Brent & Mitrovic, 2005) zkoumá inteligentní vyučovací systémy obecně. Navazuje na výzkum Mitrovic (Mitrovic, 1998) a popisuje efekt způsobený manipulací zpětnou vazbou ITS. Důraz klade na praktické využití křivek učení charakterizujících výkonnostní vlastnosti studentů. Podobný problém zkoumal i Weerasinghe (Weerasinghe & Mitrovic, 2005), který do inteligentních vyučovacích systémů implementoval simulaci strategie učitelů. Brent (Brent & Mitrovic, 2005) zdokonalil funkčnost tutora seskupením jeho existujících znalostních elementů do pravděpodobnostních konceptů. Existující modely výkonu studentů implementoval tak, aby měřily validitu nového konceptu. Na základě výsledků tutor korigoval svou zpětnou vazbu, a tím dosáhl optimálního učení studenta. Základem vnitřní reprezentace modelu studenta byla křivka učení. Adaptivní systém provedl její složitou analýzu, protože interakce se systémem tvoří pouze malou část vzdělávacích aktivit studenta. Křivka učení reprezentovala kvadratickou závislost poměru nesprávných odpovědí vůči celkovému počtu předložených problémů. U křivky učení se předpokládá nepřímá úměrnost mezi počtem předložených problémů a počtem nesprávných odpovědí. Křivka učení může být zaznamenána pro každé pravidlo systému, což umožňuje měřit jeho výkon. Jednotlivé výsledky Brent agregoval do skupin na základě taxonomie učební domény.



Obrázek 11 Architektura systému SQL-Tutor

Jednou z prvních prací Mitrovic (Mitrovic, 1998) na poli inteligentních vyučovacích strojů je její článek prezentující SQL-Tutor. To je inteligentní vyučovací systém pro programování

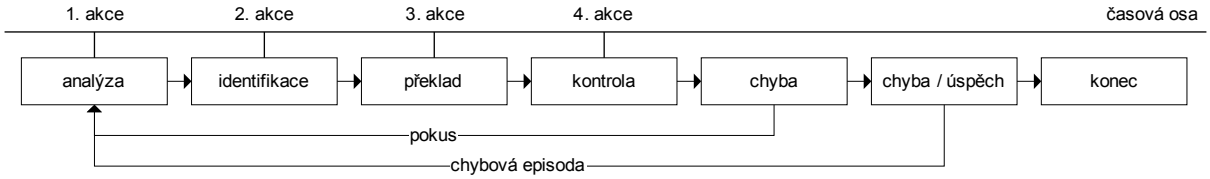
v SQL. SQL-Tutor je založen na myšlence řízeného objevování učebních prostředí a podporující řešení problému a meta-učení. SQL-Tutor umožňuje pět úrovní zpětné vazby, a to pozitivní/negativní, označení chyby, nápovědu, dílčí řešení a kompletní řešení. Systém uchovává vnitřní reprezentaci modelu studenta a používá omezující podmínky založené na záznamech o výkonu studenta. Mitrovic popisuje architekturu systému. Uvádí, že modelování a generování pedagogických akcí je založeno na individuálním přístupu ke každému studentovi. Primární určení SQL-Tutora je primárně pro poskytnutí základní praktické zkušenosti studentům. Není tedy náhradou klasického stylu vzdělávání, ale jeho doplňkem. Při učení SQL se studenti potýkají s různými druhy problémů. Některé mají svůj původ v obtížích učit se nazpaměť databázové schéma. Tím dochází k chybám nesprávného určení jména tabulky nebo atributu. Jiné chyby mají svůj původ v nesprávném pochopení elementů dotazovacího jazyka a relačního datového modelu obecně. Další pramení z nepochopení konceptů seskupování a omezeného seskupování. Ty jsou těžké díky své složitosti. Další dva běžné zdroje obtíží jsou spojování tabulek a rozdíl mezi agregační a skalární funkcí.

Jednou z efektivních metod výuky dotazovacího jazyka je nácvik ve třídě. Problémy jsou řešeny na tabuli a jsou doplněny o laboratorní cvičení. Studenti zjišťují obtíže spojené s učením SQL přímo při práci s databází, protože informační a chybové zprávy poskytují zpětnou vazbu pouze v oblasti syntaxe. Oproti tomu SQL Tutor kontroluje i sémantickou správnost odpovědí studenta. Systém je založený na principu řízeného objevování. To je jeden z výukových stylů běžně používaných v ITS aplikacích. Princip řízeného objevování vychází z přesvědčení, že studenti raději objevují věci samostatně, než aby pouze pasivně přijímali nové informace. Výhoda přístupu je v trvanlivosti znalosti, nevýhoda je dlouhá doba počáteční nejistoty.

Wu (Wu, 2008) sestavil přehled zajímavých prací zkoumajících chyby, které vznikají při učení dotazovacího jazyka. Reason je dělí na chybu v zápisu a omyly (chyby, kdy se akce nedaří podle toho, jak byly plánovány), „omyly, jejichž zdrojem jsou použitá pravidla“ (ve spojitosti s nesprávnou aplikací pravidla nebo špatným pracovním postupem) nebo „chyby založené na neznalosti“ (vlivem nedostatku plánování). Wu provedl jejich analýzu na základě definice analytické jednotky. Při analýze zotavení z chyb proces zkoumal ve chvíli, kdy uživatel zjistí, že chyboval, a provede korekci zjištěné chyby. Každá nová oprava chyby je novým pokusem o opravu a skládá se z jedné akce nebo série akcí. Chyba a následující pokusy o opravu chyby jsou nazývány chybovou episodou.

Trvání chybové episody má počátek v bodě, kdy uživatel zjistil chybu, a končí v bodě, kdy

uspěje nebo vzdá opravu. Každá chybová episoda zahrnuje jeden nebo více pokusů o opravu. Každý pokus o opravu může být buď neúspěšný a uživatel může pokračovat s dalším pokusem o opravu, nebo úspěšný. Neúspěšný pokus o opravu může vyústit v chybovou hlášku. Chybová episoda může být kvantifikována trváním a celkovým počtem pokusů o opravu. Pokus o opravu je kvantifikován intervalem mezi dvěma systémovými hláškami.



Obrázek 12 Chybová episoda

Nejznámější novodobý autor myšlenky programovacího jazyka přímo navrženého k výuce programování a rozvíjení kompetencí řešení problému je vědec Seymour Papert. Papert (Papert, 1980) se zaměřil na vytvoření jazyka, který by žákům usnadnil učení programování. Konstrukci nového programovacího jazyka založil na poznatcích získaných při spolupráci s Jeanem Piagetem. Převzal od něj model chování žáků jako stavitelů svých vlastních intelektuálních struktur. Všiml si, že se žáci učí mnoho věcí způsobem učení bez výuky. Papert chtěl ‚žákům jako stavitelům dát materiál, pomocí kterého by mohli ‚stavět' svůj intelekt. Proto vytvořil objekt v podobě želvy fungující podle matematických principů. Želvu potom umístil do rámce kognitivní minikultury nazvané LOGO, což je programovací jazyk zprostředkovávající komunikaci se želvou. Pomocí tohoto programovacího jazyka měli žáci zadávat želvě příkazy. Ta se potom pohybovala po obrazovce a mohla za sebou nechávat viditelnou stopu. Želvu bylo možné naučit nové příkazy pomocí základní sady příkazů a na jejich základě potom želvu učit příkazy další.

Ve své podstatě byl Papertův přístup nadčasový, neboť i dnes není tato myšlenka zcela zapomenuta. Příklad jejího oživení podává například Ueno (Ueno et al., 2006). Vychází z Papertovy myšlenky a aplikuje ji na vytvoření trojrozměrné animace, ve které podle popisu pohybu základními elementy, jako jsou příkazy pro krok, otočení apod., sestavuje složitější pohyby a poskytuje funkce pro konstrukci složitějších prostředí.

BASIC a LOGO jsou dva nejznámější jazyky určené pro programátory začátečníky. Zatímco BASIC (Wikipedia, 2008) má své kořeny v procedurálních jazycích (především Fortran a JOSS), kořeny jazyka LOGO se nacházejí v programovacím jazyce LISP (tedy neprocedurálním jazyku). O psychologii používání programovacího jazyka BASIC zajímavě pojednává Mayerův článek (Mayer, 1979). Kromě jazyka BASIC a LOGO bylo vytvořeno

mnoho dalších jazyků určených pro nezkušené programátory. Jedním z nich je například programovací jazyk Karel, ve kterém programátor zadává příkazy robotovi, a řídí tak jeho pohyb po obrazovce v rámci města. Město je prostředí vytvořené ze dvou základních prvků – zdí omezujících pohyb robota a značek umožňujících interakci s okolím prostřednictvím jejich sebrání nebo položení. Také Tomek (Tomek, 1982) se zabývá problematikou speciálního programovacího jazyka určeného výlučně pro úvodní seznámení s programováním. Jedná se však jen o další z mnoha mutací programovacího jazyka Karel pod názvem Josef. Otázkám učení programování dospělých se věnuje mimo jiné i Schibeci (Schibeci, 1990). Svůj kvalitativní výzkum zaměřil na čtyři skupiny studentů. Do prvních dvou skupin zařadil budoucí učitele a do dalších dvou učitele již praktikující. Závěry jeho výzkumu dokazují pozitivní přínos učení programovacího jazyka LOGO na postoj k počítačům a na sebereflexi u dospělých studentů.

Po úspěchu programovacího jazyka LOGO se objevilo mnoho podobných programovacích jazyků. Objektové programování se začalo viditelně prosazovat na počátku devadesátých let dvacátého století. Nástup objektového přístupu se odrazil i ve vývoji programovacích jazyků určených k výuce programování. Jedním z jazyků vyvinutých za účelem výuky objektového programování je simulační systém Playground. Podrobněji o něm pojednává například Fenton (Fenton & Beck, 1989). Další příklad programovacího jazyka určeného ke snadnějšímu učení programování představuje jazyk HYPERLOGO. Zprávu o jeho vývoji podává ve svém článku Yamamoto (Yamamoto et al., 1991). Základní myšlenkou HYPERLOGA je požadavek na jednoduchý jazyk s možností implementace objektového přístupu.

Pokročilým použitím programovacího jazyka LOGO ve vyučování se zabývá Yehoshafat (Yehoshafat, 1991). Soustřeďuje se na výuku konceptu paralelního rekurzivního programování. Místo prostředí s jednou želvou využívá prostředí s několika želvami, a zkoumá tak implementaci paralelních rekurzivních algoritmů. Počítačové jazyky určené k výuce programování mohou pomoci při vývoji správného funkčního modelu počítače. Na rozdíl od programovacího jazyka Pascal může programovací jazyk LOGO umožnit vytvoření kognitivního modelu fungování počítače založeného na pohybu želvy a kreslení po obrazovce. Výhodou programovacího jazyka Karel je zase možnost poznávat základní struktury jazyka Pascal (sekvence, rozhodování, smyčky) bez znalosti použití datových struktur.

O zajímavém spojení programovacího jazyka LOGO a stavebnice LEGO napsal článek Resnick (Resnick, 1993). Popsal postupné rozšiřování stavebnice o další, interaktivní prvky. Zatímco v první generaci stavebnice bylo umožněno žákům stavět struktury, ve druhé již bylo

možné vytvořit mechanismus. Ve třetí fázi pak přibyla možnost sestavit chování. Díky tomu bylo možné sestrojovat behaviorální mechanismy, jako je například robot, který chce jít za světlem. První stavebnice LEGO Mindstorms, která je přímo založena na idejích publikovaných Papertem v knize Mindstorms (Papert, 1980), se na na trhu objevila v roce 1998. Tématem implementace jazyka LOGO ve stavebnici LEGO se ve svém článku zabývá také Enkenberg (Enkenberg, 1994). Do dnešní doby se myšlenka počítačových jazyků a strojů určených pro výuku programování zachovala jako různé systémy a stroje pro podporu učení. Pro tuto práci jsou především relevantní tzv. inteligentní vyučovací systémy, které slouží jako podpora výuky dotazovacího jazyka, návrhu struktury databáze pomocí ER modelu nebo normalizace databáze. Podrobněji se tomuto tématu věnuje (Mitrovic, 1998), (Suraweera & Mitrovic, 2002), (Brent & Mitrovic, 2005) aj.

Počátek sériové výroby mikropočítačů a jejich následné masivní rozšíření do domácností tvoří v historii počítačů významné milníky. Z hlediska tématu práce je podstatný také posun programování od složité disciplíny určené inženýrům pracujícím u sálových počítačů k disciplíně pro mnohem širší skupinu uživatelů. Důsledkem tohoto posunu je pokus o vytvoření programovacích jazyků určených přímo k výuce principů programování (např. LOGO, PASCAL nebo KAREL). Důležitý poznatek představuje i fakt, že zatímco LOGO má své kořeny v neprocedurálních jazycích, jako je LISP, BASIC, vychází z jazyků procedurálních, jako je Fortran. Na základě tohoto kritéria se rozlišují dva hlavní směry ve vývoji propedeutických programovacích jazyků – první (např. LOGO) vychází z principů neprocedurálního programování, zatímco druhý (např. KAREL) staví na principech programování procedurálního.

4.5 Ukázka obsahu výuky databázového myšlení

Existují různé názory, jak přistupovat k rozvoji databázového myšlení. Jedním z možných přístupů je vyučovat metody databázového návrhu. To doporučuje Chrisman (Chrisman, 1982) za použití entitně-relačních nástrojů. Obdobně Meeker (Meeker & Nohl, 2007) publikoval názor, že pasivní učební aktivity jako by měly být doplňovány aktivními praktickými aktivitami. U studentů by to umožnilo vyšší sebejistotu a rozvoj jejich kompetencí v oblasti práce s neznámým softwarovým systémem, změny existujících programů a sledování vlivu jejich akcí na systém. Výše uvedené dovednosti by mohly být rozvíjeny, aniž by se museli studenti strachovat o negativní dopad na počítačový systém, který používají. Caldeira (Caldeira, 2008) se věnuje inovacím metodik výuky dotazovacího jazyka. Domnívá se, že dříve, než bude po studentech požadováno psaní dotazů, je nutné je

naučit dotazy naučit číst. Studenti na tuto změnu reagovali pozitivně. Hollingsworth (Hollingsworth, 2008) se domnívá, že výuka psaní SQL dotazů by měla být prováděna pomocí informovaného instruování (místo tradičního instruování) a při použití této metody pozorovat zlepšení ve výkonu studentů při psaní dotazů. Tradiční instruování podle Brunerova probíhá následovně: (T1) student sleduje, jak instruktor demonstruje strategii, (T2) studentovi je dovoleno procvičit si strategii, (T3) studentovi není podáno vysvětlení, proč strategie fungovala. Nakonec (T4) studentovi není podána zpětná vazba na jeho výkon, když strategii procvičuje. Oproti tomu informované instruování probíhá takto: (T1) student sleduje, jak instruktor předvádí strategii, (T2) je mu dovoleno strategii procvičit, (T3) je mu podána okamžitá zpětná vazba, jak úspěšný byl při jejím používání. Nakonec je mu podáno (T4) vysvětlení, proč strategie fungovala a (T5) za jakých podmínek má aplikovat strategii.

Výzkum metodik výuky programování vychází ze studií porovnávajících počítačové jazyky. Weltyho (Welty & Stemple, 1981) experiment dával do souvislosti schopnost zvládnout jazyk se stupněm jeho procedurálnosti. Jarke (Jarke & Vassiliou, 1985) pro změnu zkoumal otázku výběru vhodného dotazovacího jazyka. Stanovil metodu vzájemného porovnání dotazovacích jazyků ve stanovených kategoriích. Při srovnání se nezaměřil na konkrétní produkt, ale jazyky srovnával poměrem ceny a výkonu. Jeho článek je důležitým zdrojem informací o historii a vývojových trendech v dotazovacích jazycích. Ve svém článku se věnoval kategorizaci dotazovacích jazyků a pro účely pozorování znalosti sestavil model interakce v dotazovacím jazyce. Podle Priora (Prior, 2003) má výuková strategie formulace dotazů bez jejich zadávání do databáze vliv na učební strategii studentů v daném předmětu. Kearns (Kearns et al., 1996) kategorizoval výukové metody databázových systémů. Rozdělil je z hlediska informačního systému (s důrazem na datové modelování), softwarového systému (s důrazem položeným na analýzu a ladění výkonu) nebo programovacího jazyka (především logické uvažování a optimalizace). U výuky zkoumal stupeň vhodnosti prezentace modelu vyhodnocování dotazu databázovým strojem. Mayer (Mayer, 1979) ve svém výzkumu výuky programovacího jazyka doporučil:

1) vyučovat základní transakce obsažené včetně lokací, objektů a operací,
2) vyučovat sekvence transakcí pro každý předpříkaz,
3) rozlišovat mezi různými prestatementy, které sdílejí stejné jméno příkazu,
4) vysvětlovat povinné úseky programu,
5) vysvětlovat nepovinné úseky programu,
6) nechat studenty vyhodnotit, které transakce budou provedeny v předloženém kódu,
7) zdůraznit techniky pro generování podprogramů a strukturovaného programování.

Bottino (Bottino, 1992) zjistil srovnáním učení procedurálního a deklarativního programovacího jazyka, že rozhodnutí o použití konkrétního programovacího jazyka ve výuce má záviset především na budoucí možné aplikaci a má brát v potaz pedagogické záměry. Z výsledků jeho výzkumů vyplývá, že deklarativní programovací jazyk umožňuje lépe dosáhnout pedagogických cílů výuky reprezentace znalostí dané domény, logického přístupu k řešení problémů, konstrukce databází a manipulace a práce s dotazy. Ma (Ma, 1994) se zabývá otázkou vyučovacího prostředí, které by mohlo stimulovat zájem studentů a podpořit je v zapojení do celého výukového procesu. Toho chce dosáhnout aplikací metod problémového učení. Ve svém článku Ma představuje experiment, ve kterém byl studentům představen praktický problém. Ten byl formulován do podoby zadání projektu. Studenti na jeho základě měli navrhnout a implementovat databázový systém, který by praktický problém vyřešil. Byli rozděleni do skupin a zároveň se učili dovednosti a znalosti, které nejsou vyučovány ve třídě. Pro vyhodnocení výsledků problémového vyučování byla použita metoda formálního rozhodování. Ta dovolila studentům podílet se na procesu evaluace a vytvořila vhodné prostředí pro učení díky aplikaci spravedlivé metody hodnocení. Přístup studentů k učení byl analyzován dotazníkem. Statistika výsledků ukázala, že existuje lineární vztah mezi hloubkou přístupu k učení a výsledkem projektu.

V kompetenční rovině lze na rozvoj databázového myšlení pohlížet jako na rozvoj znalostí a dovedností nutných pro formulaci dotazů ve strukturovaném dotazovacím jazyku. Vhodné uspořádání je založeno na poznání dílčích prvků a směřuje k cílovým kompetencím databázového myšlení. Udoh (Udoh, 2006) navrhuje vyučovat databázové myšlení v reálném prostředí databázového systému. To by mělo mít pozitivní vliv na výsledek učení studentů. Vědci se v rámci výuky počítačových jazyků věnovali výzkumu rozdílného vlivu vyučovacích strategií a pořadí učení jednotlivých syntaktických a sémantických prvků. Jedním z příkladů definování jasného principu rozvoje zvládnutí počítačového jazyka nastínil Shneiderman (Shneiderman et al., 1977). Popsal spirálovou metodu vyučování založenou na prezentaci syntaktické a sémantické znalosti po malých úsecích tak, aby odpovídaly myšlenkové struktuře znalostí žáka. Jeho strategie začínala s výukou programovacího jazyka na úrovni vstupních a výstupních příkazů. Následně postupoval od nejjednodušší formy přiřazování příkazů k tvorbě aritmetických výrazů. Každý následující krok by měl obsahovat koordinovaný rozvoj syntaktických a sémantických prvků znalostí. Nová znalost by měla být vztažena k předchozí znalosti a měla by být okamžitě demonstrována na smysluplných a relevantních příkladech. Kromě toho by mělo být studentovi umožněno ihned novou znalost procvičit. Model poskytuje zpevnění již osvojených materiálů a rozvíjí sebedůvěru úspěšným

řešením postupně složitějších úloh. Shneiderman zároveň hledal spojnici ve vztahu mezi strukturovaným programováním a vývojovými diagramy.

Výuka s aplikovaným spirálovitým přístupem však klade větší požadavky na učitele. Zvýšená zátěž na učitele je v podobě nezbytné času věnovanému přípravě strukturovaného uspořádání učebních textů. Netriviální části musejí být vynechány, aby bylo studentům předloženo minimum použitelných podmnožin jazyka, a ty jsou následně postupně rozšiřovány. Studenti musí zároveň pravidelně procvičovat problémové úlohy a učitel kontrolovat stav jejich znalostí. Specifikum spirálového přístupu spočívá v jeho efektivitě pro učení nejslabších studentů. Jinou výukovou metodu použitelnou ve výuce databázových systémů popisuje Ma (Ma, 1994). Provedl experiment, ve kterém se pokusil metodami problémového vyučování zvýšit zájem studentů o problematiku databázových systémů. Jiný přístup zvolil Gibson (Gibson & O'Kelly, 2005), který publikoval model založený na softwarovém inženýrství. Ten použil, aby odůvodnil kognitivní vazbu mezi řešením problému a učením programování. Při výuce databázového myšlení lze ovšem použít i heuristickou metodu. Ta patří mezi aktivizující vyučovací metody (Maňák & Švec, 2003). Na rozdíl od jiných metod učitel žákům nesděluje poznatky přímo, ale vede je k tomu, aby si je samostatně osvojovali. Hlavní techniky heuristické metody jsou založeny na vyučování podporujícím objevování, pátrání a hledání řešení problému. Takové strategie a techniky motivují a pomáhají k rozvoji potřebných dovedností.

Hlavním problémem při rozvoji didaktických prostředků pro oblast rozvoje databázového myšlení je úloha sestavení aparátu, který dovolí zpracovat odpovědi žáků. Jeden z možných přístupů k návrhu systému publikoval Whittaker (Whittaker & Stenton, 1989). Provedl studii a vyhodnocení různých návrhu systémů určených pro zpracování přirozeného jazyka. Zároveň vypracoval metodu pro jejich evaluaci. Zvláštnost jeho metody spočívá v její nenáročnosti na technické prostředky. Cílem bylo nechat subjekt sestavit podle zadání bez pomoci počítače dotaz v přirozeném jazyce. Použitá metoda ve výsledku umožnila sledovat myšlenkové pochody a požadavky použitelné při implementaci dotazovacího jazyka. Obdobná metoda se jmenuje ‚Kouzelník ze země OZ', ve které vybraný člověk simuluje funkcionalitu konstruovaného systému.

Soloway (Soloway, 1986) považoval za hlavní cíl učení konkrétního počítačového jazyka konstrukci programu. Navrhnul posun od tradičních metod programování založených na učení syntaxe a sémantiky k výuce knihoven funkcí, ze kterých by studenti měli sestavit program. Výuka programování je na některých školách považována za pracovní dovednost, a proto se ji nepokusil zahrnut do obecného kurikula. Peelle (Peelle, 1983) se zabýval přístupem

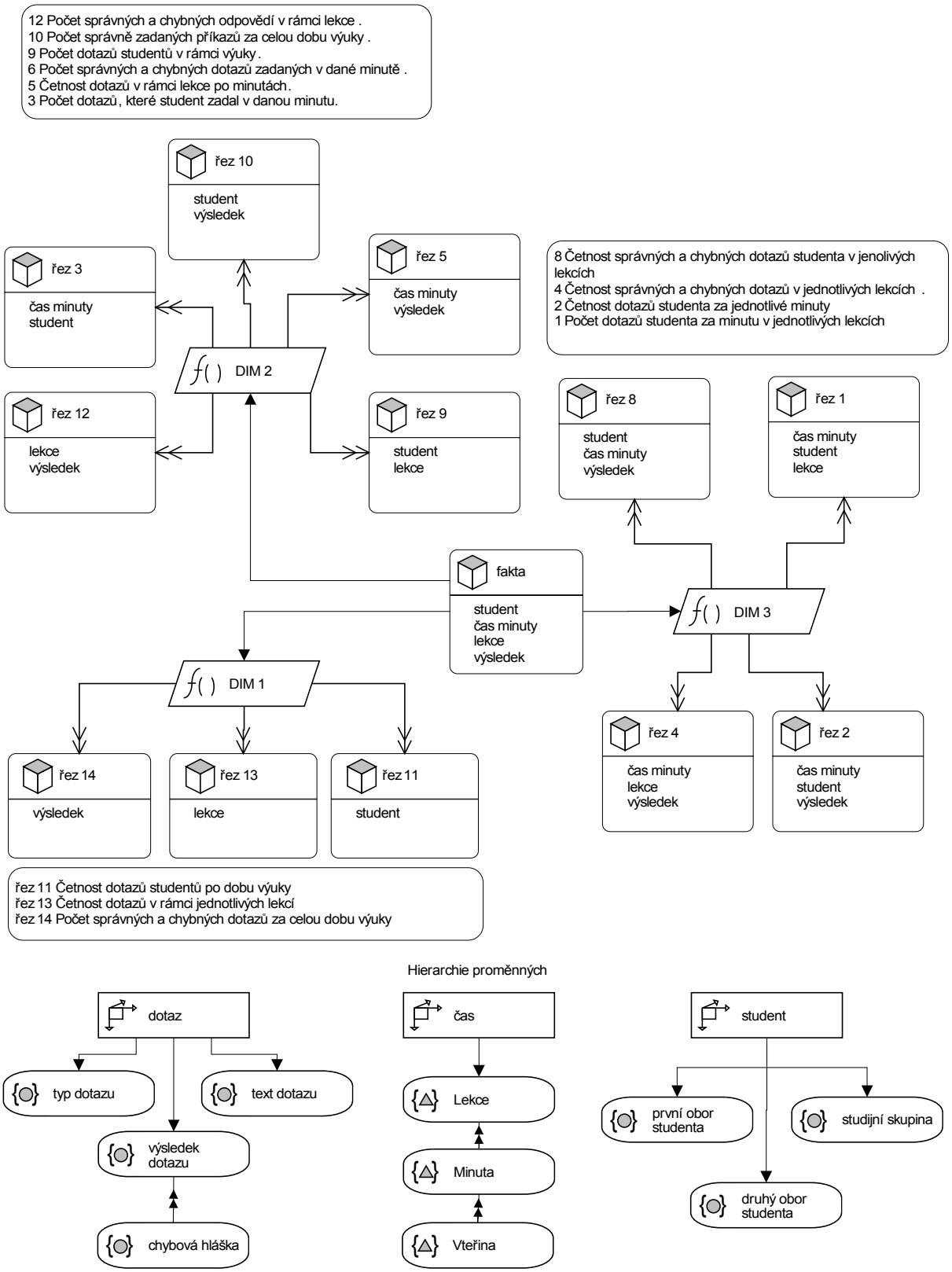
umožňujícím učitelům lépe využít počítačovou gramotnost ve výuce. Jednalo se o metafory, které učitelům pomohly pochopit koncepty informační gramotnosti. Metafory autora článku přirovnávají počítač k osobě, nástroji, mozku, mapě, průhledné skříňce, vitamínu, paletě a učiteli.

5 Evaluace efektivity rozvoje databázového myšlení

Cílem praktické části bylo určit kvantitativní charakteristiky rozvoje databázového myšlení. Konkrétně určení a rozpracování metrik rozvoje schopnosti studenta správně formulovat dotaz ve strukturovaném dotazovacím jazyce (SQL). Jako výsledek praktické části byla na základě provedených analýz stanovena metodika analýzy rozvoje databázového myšlení. V procesu stanovení metrik a jejich analýzy byly jako hlavní vstupní proměnné použity četnost dotazů a chybovost studentů ve výuce. Při syntéze výsledků existuje možnost navržení experimentu, který by mohl blíže zkoumat efektivitu použitého grafického rozhraní, blíže analyzovat vliv vstupních výsledků na rozvoj databázového myšlení, podrobněji analyzovat charakter vztahu mezi četností a chybovostí studentů nebo prozkoumat rozdíl ve výkonech studentů. Zároveň by bylo možné použít v analýze kvalitativních metod, které by blíže vysvětlily zdroj chybovosti studentů nebo řízeným interview mohly určit další faktory rozvoje databázového myšlení.

Údaje, jež byly podkladem pro analýzu, pocházely z výuky vysokoškolského kurzu databázových systémů. Ten probíhal na Pedagogické fakultě Univerzity Karlovy a účastnili se jej posluchači studia učitelství ICT.

Základní soubor dat tvořil záznam výuky. Při ní byly na databázovém serveru zaznamenávány SQL dotazy studentů. S každým uloženým SQL dotazem byla zároveň uložena informace o tom, který student ji zadal, přesný čas vykonání, typ příkazu a hlášení serveru o výsledku dotazu. Na základním datovém souboru byla provedena identifikace kvantifikovatelných charakteristik. Hlavní charakteristika vyjadřující stupeň rozvoje databázového myšlení studenta byl počet jeho dotazů vykonaných databází s ohledem na jejich správnost. V kontextu cíle praktické části bylo nezbytné analyzovat nejenom kvantitu a správnost činností studenta ve výuce, ale i určit statistické metody pro jejich zpracování.



Obrázek 13 Datový model statistického souboru

Experiment by bylo vhodné realizovat kontinuálně a ve větším měřítku. S ohledem na velikost souboru dat nelze všechny výsledky zobecňovat. Mohou však poskytnout výstup pro

hlavní cíl práce a tím je stanovení možných charakteristik učení dotazovacího jazyka. Aby bylo možné z výsledků vyvodit konkrétní vzory chování studenta při učení dotazovacího jazyka, bylo by nutné provést analýzu ve větším měřítku. Kromě toho vidím jako možnost provést ověření hypotézy rozdělením studentů do dvou různých skupin a v každé z nich aplikovat jiné vyučovací metody s cílem hledání té optimální.

Před zahájením výuky studenti první hodinu vyplnili vstupní dotazník, jehož cílem bylo zjistit vstupní úroveň studentů. Ten byl potom využit k hledání charakteristik, jež by mohly poskytnout souvislost mezi vstupními předpoklady a výsledkem učení.

Součástí výuky na Katedře informačních technologií a technické výchovy je i předmět „Databázové a informační systémy“. Jedná se o povinný předmět pro všechny studenty oboru Technické a informační výchovy. Rozsah předmětu činil 1 h přednáška/1 h seminář týdně. Studenti za absolvování předmětu obdrželi 3 kredity. Studijní předmět byl ukončen klasifikovaným zápočtem.

Celková výuka předmětu byla rozdělena na tři tematické celky:
Databázové systémy – tříúrovňová ANSI-SPARC architektura, datové modely, relační model: schéma databáze, entity, atributy, relace, kardinalita, klíče, diagramy RD (Realations Diagrams), normalizace, architektura zpracování a přístupu k datům: host-terminál, file-server, klient server, lokální distribuované databáze.

Informační systémy – pojmy (systém, soustava, komplexní přístup, systémový přístup), příklady informačních systémů (podnikové, osobní, státní, školní, knihovní, GIS).

Vývoj informačního systému – životní cyklus projektu, modely životního cyklu projektu, metody analýzy, návrhu, testování, CASE systémy

Práce s databázovým systémem – pojmy a architektura: model dat, typy položek, prostředí, práce s databází: práce s existující databází: pořízení, editace, prohlížení, hledání, třídění, tisk, vytváření databáze: návrh tabulek (návrh položek, návrh relací mezi tabulkami), dotazy, tvorba formuláře, tvorba sestavy.

Jednotlivé vyučovací hodiny měly následující obsah:

Lekce	Téma
1	Vývoj přístupu ke zpracování dat, organizace dat v agendovém zpracování, databáze jako nástroj zpracování dat, systém řízení báze dat, datové modely.
2	Relační datový model, 12 pravidel Codda, objektově orientovaný přístup.
3	Přihlášení do databáze, nastavení znakové sady, nastavení řazení záznamů, vytvoření tabulky čítající jeden sloupec, zobrazení popisu tabulky, základy změn struktury tabulky.
4	Příkazy pro definici dat DDL, práce v textovém rozhraní. Vytváření tabulky, zobrazení a změna struktury tabulky, vkládání dat do tabulky a jejich odstranění, přejmenování tabulky, mazání tabulky.
5	Vlastnosti systému MySQL, architektura klient server, proces návrhu databáze, klíč - relace 1:1, relace 1:N, relace M:N, indexy (create, drop index), datové typy, auto_increment, zerofill, textové datové typy, vyhrazená slova jazyka SQL.
6	Teorie: entita, vztah, kardinalita. Praxe: vytváření tabulky, vkládání a získání dat z tabulky, mazání tabulky.
7	Teoretické základy používání grafického nástroje pro práci s databází.
8	Procvičování grafického nástroje pro vytváření, mazání a manipulaci s tabulkami.
9	Základní a pokročilá syntaxe příkazu select.
10	Konceptuální modelování, typy relací, spojování tabulek.
11	Procvičování spojení tabulek.
12	Odevzdání a kontrola seminární práce.
13	Udělování zápočtů.
14	Předtermín.

Tabulka 5 Témata lekcí kurzu Databázové systémy

V rámci každé lekce bylo probráno téma a po jeho výkladu následovalo jeho procvičování zadáváním dotazů do databázového systému. Každý student měl vlastní uživatelský účet. To později umožnilo analýzu výkonu každého studenta. Celou výuku lze z hlediska použitého technického prostředku rozdělit do dvou časových pásem. V prvním bylo studenty používáno textové rozhraní a ve druhém studenti používali grafické rozhraní.

Při analýze chybovosti bylo základním úskalím správně stanovit interval pro výpočet četnosti charakteristiky. Při kvantifikaci četnosti v intervalu vyučovací hodiny obsahovala charakteristika chybovosti studenta dvanáct hodnot. Pokud byl naopak zvolen interval jedné minuty, nenabývala četnost dostačujícího počtu. Za optimální se proto jevil interval deseti minut, který poskytoval dostatečný objem dat s vypovídající četností.

Po ukončení sběru hodnot byla provedena datová normalizace vstupních údajů, určení primárních klíčů a návrh klíče pro spojení údajů ze vstupního dotazníku s údaji zadávanými v průběhu výuky. Výsledkem byl statistický soubor, jehož diagram uvádím.

Jak bylo uvedeno výše, studenti měli na počátku výuky za úkol vyplnit vstupní dotazník. Ten se skládal z otázek, jejichž cílem bylo zjistit zkušenosti studentů s počítačem a jejich možnosti práce s počítačem. Otázky byly koncipovány do oblastí dostupnosti počítače, délky používání počítače, primárního případu použití počítače, zkušeností práce s počítačem, zdroje poznání práce s počítačem a znalosti databázových produktů. Celkem bylo do zpracování zařazeno 18 dotazníků. Dotazník se skládal z uzavřených otázek obsahujících 10 otázek.

Č. Otázky	**Otázka**
1	Počítač mám pro sebe dostupný
2	Počítač používám již
3	Počítač používám především pro
4	Dobře se vyznám a umím
5	O mých zkušenostech s počítačem lze říci
6	Kdo Vás naučil nejvíce z toho, co víte o počítačích a jejich používání?
7	Kdo Vás naučil nejvíce z toho, co víte o internetu a jeho používání?
8	S databázovými systémy
9	Již před tímto kurzem jsem znal(a) tyto databázové produkty
10	V předmětu Databázové a informační systémy bych se chtěl

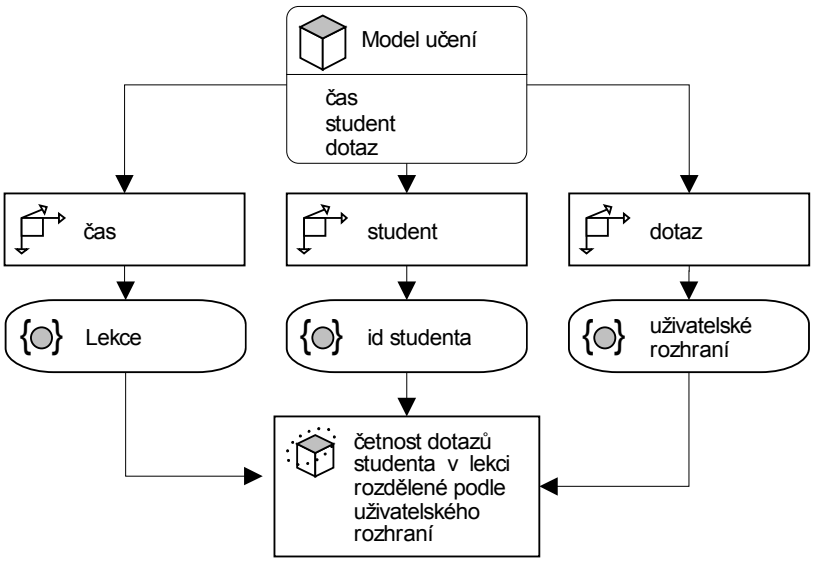
Tabulka 6 Vstupní dotazník (odpovědi viz Tabulka 34 Vstupní dotazník)

V praktické části byly pro analýzu rozvoje databázového myšlení ve výuce použity statistické metody. Především to byl Kolmogorov-Smirnův test, základní statistické charakteristiky, korelace četnosti a chybovosti dotazů, regresní analýza chybovosti a četnosti dotazů a analýza rozptylu.

5.1 Analýza četnosti dotazů

5.1.1 Metodika výpočtu

V rámci analýzy rozdílu mezi výkonem studenta v textovém a grafickém rozhraní byly jako vstupní proměnné vybrány počty dotazů studenta v lekci. Koncepce analýzy směřuje k zodpovězení otázky, zdali se odlišoval počet a chybovost v jednotlivých rozhraních od sebe navzájem. Pro komparaci obou metod zadávání SQL dotazů byly použity testy normality statistického souboru s cílem ověřit. Na základě jejich výsledku vyplynula možnost použití t-testu.



Obrázek 14 Model souvislosti a hierarchie proměnných

Aby bylo možné provést komparaci použitých uživatelských rozhraní, bylo nezbytné stanovit stupnice a spojitost jednotlivých proměnných. Dále potom vybrat vhodnou granulitu zkoumaných proměnných.

Kategorie	Jméno proměnné	Stupnice	Spojitost
Čas	Lekce	nominální	diskrétní
Čas	Měsíc	ordinální	spojitá
Čas	Týden	ordinální	spojitá
Čas	Den	ordinální	spojitá
Čas	Minuta	ordinální	spojitá
Čas	Vteřina	ordinální	spojitá
Student	id studenta	nominální	diskrétní
Dotaz	text dotazu	nominální	diskrétní
Dotaz	výsledek dotazu	nominální	diskrétní
Dotaz	chybová hláška	nominální	diskrétní
Dotaz	uživatelské rozhraní	nominální	diskrétní

Tabulka 7 Proměnné vybrané k bližšímu zkoumání

Na základě analýzy granulity proměnných byla jako vhodný kandidát vybrána úroveň počtu SQL dotazů položených v rámci jedné vyučovací lekce. Podle výběru byly kvantifikovány četnosti dotazů pro jednotlivé lekce.

	Lekce								
Student	03	04	05	06	07	08	09	10	11
19	1	19	51		33	495	536	387	359
20		16	40		23		357	698	297
21			41		11	489	551		386
22			49		10	214	215	563	274
23		21	32			418		676	283
24	1	16	56			256	438	410	361
25		20	36			250		379	283
26	14	23	124	297		455	522	424	104
27	1	40	75	47	14	174	283	232	892
28		31	53		14	216	254	468	274
29	2	5	75			930		716	829
32	7	71	40		23	370		943	
33	4	43				207	350	595	644
34	19	30	37		10	339	413		433
35	19	33	28		9	203		762	398
36	17					465	254	521	
37		54	47	9	11	473	598	313	455
38		23	57		30	685		578	635
39		56			19	239	315	473	271
40		29	59				1055	815	506
41		3	34			231	352	236	188
42		29	39		16	516	585	507	307
43		52		6	23	240		926	424
44		17			22	173	471	386	98
45	14								
46		32	99		18			444	843

Tabulka 8 Aktivita studentů v jednotlivých lekcích měřena počtem SQL dotazů

5.1.2 Test normality a shody průměrů

Aby bylo možné správně vybrat metody, které budou dále aplikovány na chybovost studentů, byl proveden test normality. Ten byl proveden odděleně pro chybovost v textovém rozhraní, grafickém rozhraní a celkem.

	Testy normality					
	N	max D	K-S p	Lilliefors p	W	p
Četnost	73	0,2	p < ,01	p < ,01	0,58	0

Tabulka 9 Test normality učení v textovém rozhraní

	Testy normality					
	N	max D	K-S p	Lilliefors p	W	p
Četnost	92	0,08	p > .20	p < ,10	0,97	0,01

Tabulka 10 Test normality učení v grafickém rozhraní

Vzhledem k zamítnutí hypotézy o normalitě dat o učení v grafickém rozhraní bude nutné pro srovnání obou metod použít metod neparametrické statistiky.

	Wilcoxonův párový test na hladině p <,05000			
	Počet platných	T	Z	Úroveň p
Textové & grafické rozhraní	73	14,0	7,3	0

Tabulka 11 Wilcoxonův párový test rozdílu učení v textovém a grafickém rozhraní

Wilcoxonův test vyjadřuje rozdíly mezi dvěma skupinami.

	Znaménkový test na hladině p <,05000			
	Počet různých	procent v < V	Z	Úroveň p
Textové & grafické rozhraní	73	94,5	7,49	0

Tabulka 12 Znaménkový test rozdílu učení v textovém a grafickém rozhraní

Testy variability průměrů ukázali statisticky významný rozdíl mezi průměrným počtem všech, správných i nesprávných dotazů zadaných v grafickém uživatelském rozhraní v porovnání s textovým uživatelským rozhraním.

Výsledek lze interpretovat tak, že studenti se při používání textového rozhraní zadávají menší počet dotazů. Je to způsobeno jednak tím, že musí jednotlivé příkazy zadávat ručně, a proto jsou omezeni rychlostí psaní jednotlivých dotazů. Oproti tomu v grafickém rozhraní je možné zadávat více příkazu už jenom používáním grafického rozhraní. Na druhou stranu výsledky poukazují na menší chybovost v případě použití textového rozhraní oproti použití rozhraní grafického.

5.1.3 Kolmogorov-Smirnův test

Pro test normality byl použit Kolmogorov-Smirnův test. Lze jej aplikovat na jednu proměnnou statistického souboru a je určený k vyjádření shody proměnné s hypotetickým spojitým rozdělením. Anglicky se test nazývá „The Kolmogorov-Smirnov Goodness of Fit Test“. Při jejím výpočtu je ověřována hypotéza H_0: Proměnná X má kumulativní pravděpodobnostní rozdělení:

$$F_x(x) \S F(x)$$

Nechť je $S_n(x)$ sledované kumulativní rozdělení náhodné proměnné, $x_1, x_2, ..., x_n$ také známé jako empirické rozdělení. Předpokládejme, že data jsou seřazena sestupně, hodnoty $S_n(x)$ jsou získány přidáním následující frekvenci výskytu, k_i / n pro každé různé x_i .Nulová hypotéza předpokládá získání malé odchylky $S_n(x)$ od $F(x)$. Kolmogorov-Smirnův test používá největší z těchto odchylek jako míru shody.

$$D_n = \max\left|F(x) - S_n(x)\right|, \text{ pro každé jedinečné } x_i$$

Hodnoty rozdělení D_n jsou publikovány. Pokud n není velice malé, může být použit následující asymptotický výsledek:

$$\lim_{x \to \infty} P(\sqrt{n} D_n \Ł t) = 1 - 2\left|\sum_{i=1}^{\infty} (-1)^{i-1} e^{-2i^2 t^2}\right.$$

Kolmogorov-Smirnův test zamítá nulovou hypotézu na úrovni α , pokud $D_n > d_{n,\alpha}$, kde pro $d_{n,\alpha}$ platí, že

$$P_{H_0}(D_n > d_{n,\alpha}) = \alpha$$

Při použití vzorce lze získat následující kritické hodnoty:

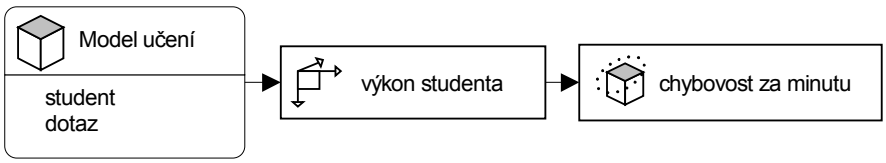
$$d_{n,0.01} = 1.63/\sqrt{n}; d_{n,0.05} = 1.36/\sqrt{n}; d_{n,0.10} = 1.22/\sqrt{n}$$

Při aplikaci Kolmogorov-Smirnova testu byly použity aktuální distribuční parametry dat. (Hendl, 2004)

5.2 Trend chybovosti studentů

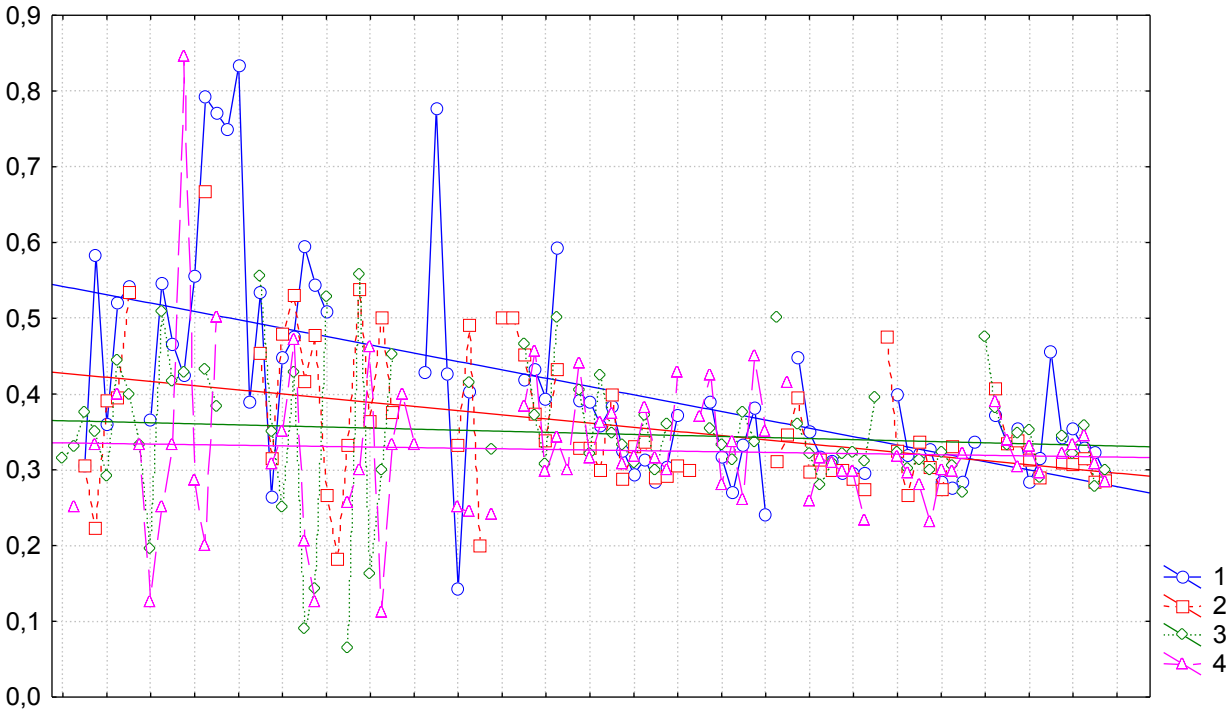
5.2.1 Metodika výpočtu

Předchozí kvantitativní charakteristiky byly zaměřeny na kvantifikaci rozdílu mezi použitím grafického a textového uživatelského rozhraní a analýzu závislosti chybovosti na počtu databázových dotazů. Dalším kvantitativním vyjádřením rozvoje databázového myšlení je sledování trendu chybovosti studentů v čase. Pro analýzu trendu chybovosti studentů byla jako stěžejní metrika pouze vybrána proměnná chybovost za určený časový úsek.

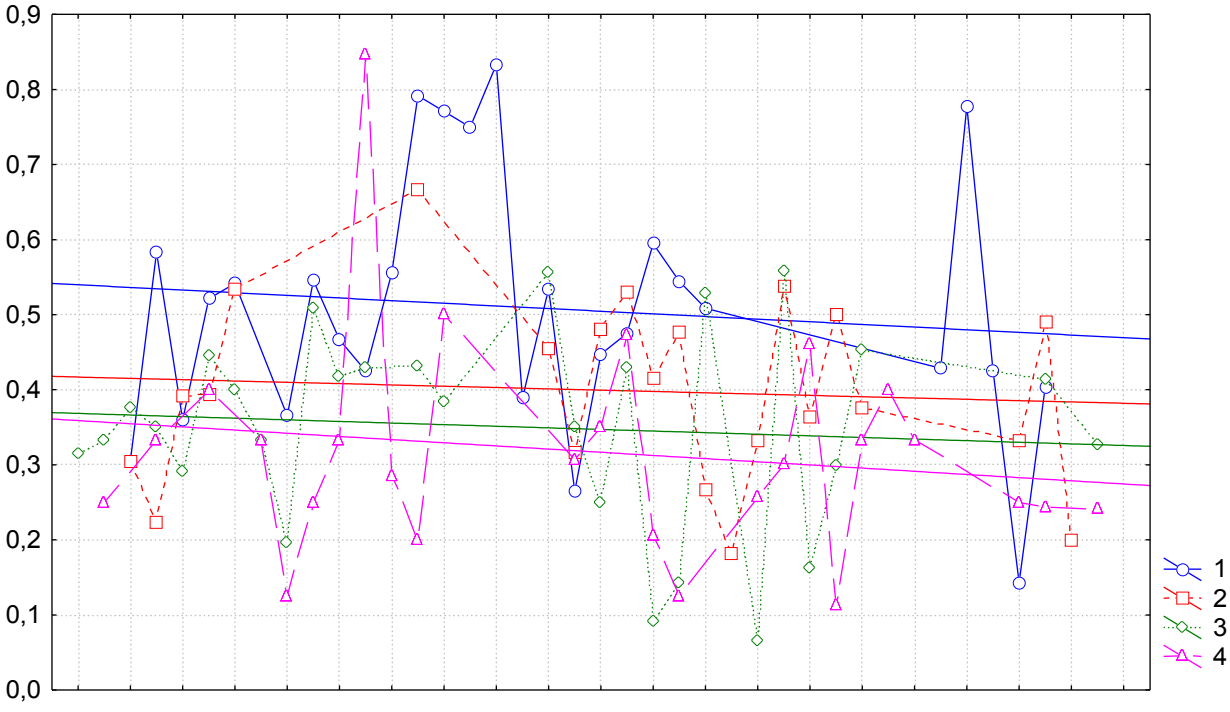


Obrázek 15 Model hierarchie vstupní proměnné

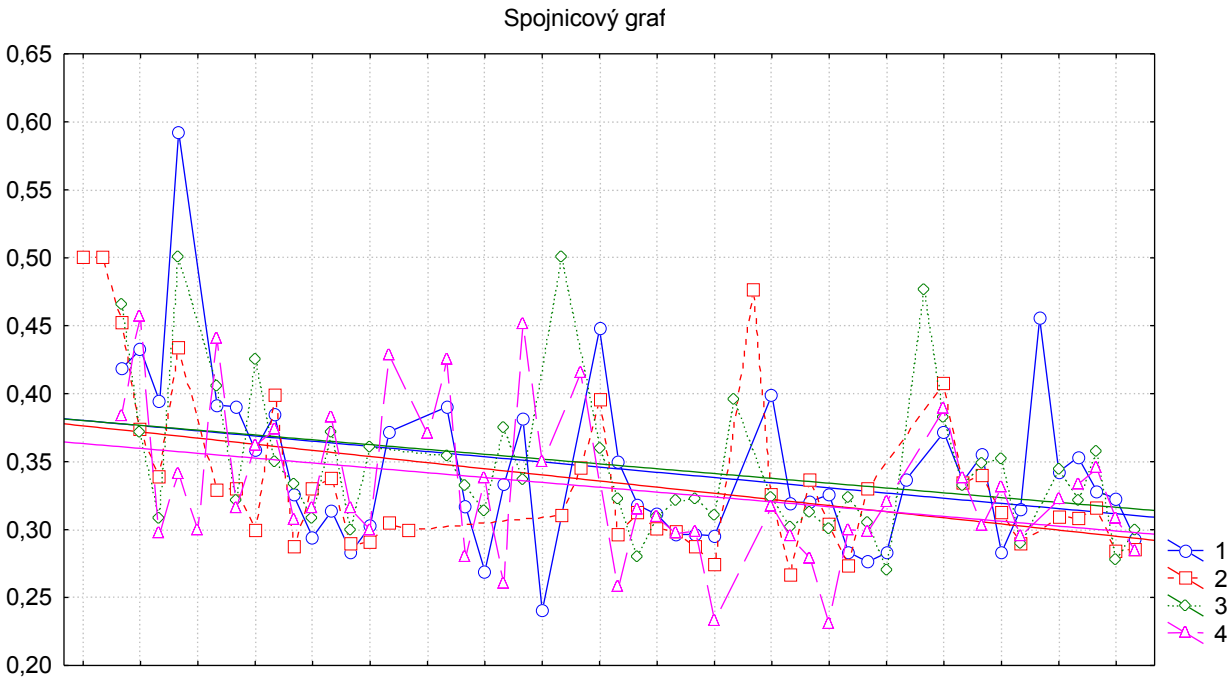
Pro sledování výkonnosti studentů byla vybrána chybovost v intervalu deseti minut. Pro lepší možnost kvantifikace chybovosti jsem provedl kategorizace výkonů studentů do čtyř skupin. Ty jsem rozdělil podle průměrné chybovosti studentů v rámci celé výuky. Výše popsané rozdělení studentů do výkonnostních skupin umožnilo lepší izolaci extrémních hodnot a dovolilo vyjádřit trend vývoje chybovosti graficky. Chybovost studenta v daný časový úsek je vyjádřena poměrem chybných SQL dotazů k celkovému počtu SQL dotazů.



Obrázek 16 Trend chybovosti studentů při výuce dotazovacího jazyka bez ohledu na GUI



Obrázek 17 Trend chybovosti studentů při výuce dotazovacího jazyka v textovém rozhraní



Obrázek 18 Trend chybovosti v rámci grafického rozhraní

Při bližším prozkoumání grafů je patrné, že sklon křivky učení v textovém rozhraní je méně strmý než sklon křivky učení v grafickém rozhraní. Z tohoto rozdílu lze pozorovat rozdílný trend sestupné chybovosti.

5.2.2 Výsledky analýzy trendu chybovosti

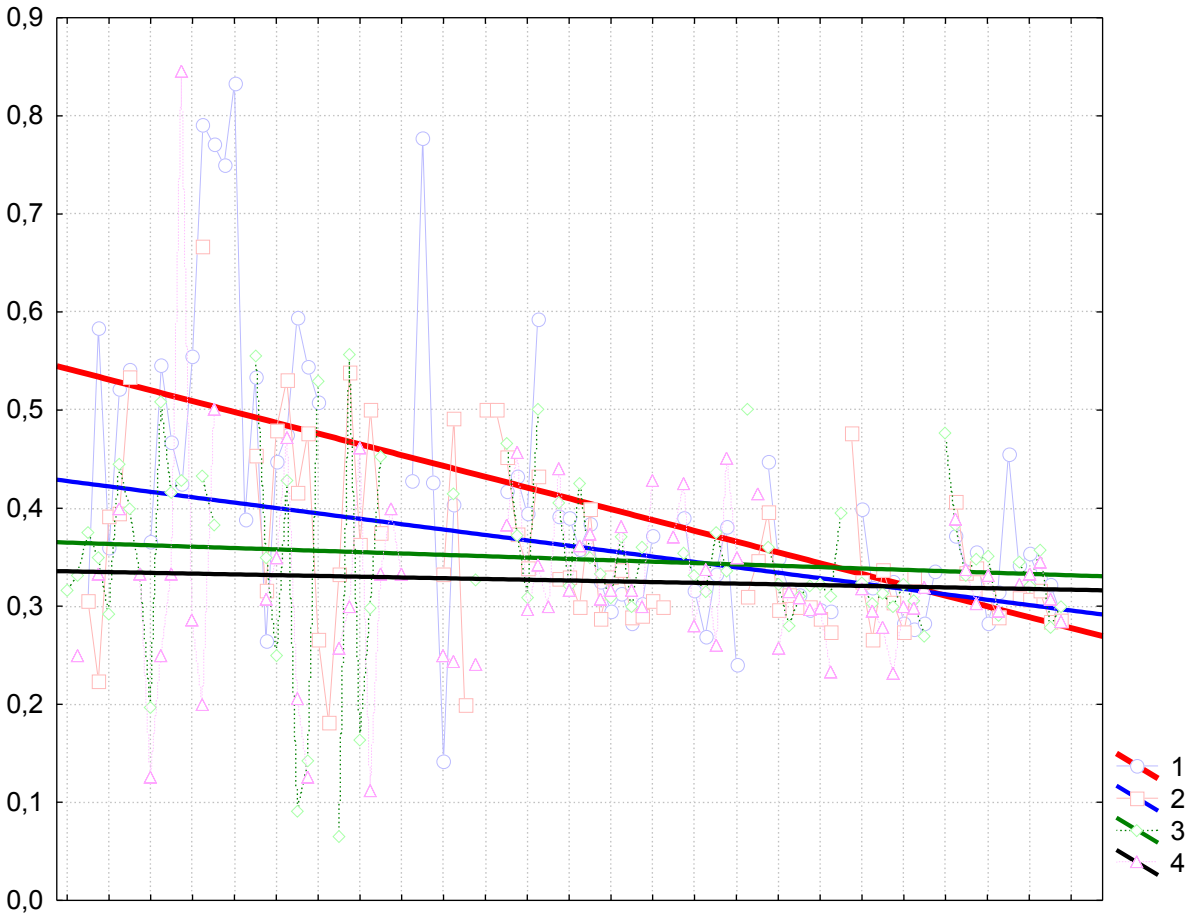
Hlavním cílem kvantifikace trendu chybovosti bylo vyjádření stupně rozvoje databázového myšlení. Potažmo zlepšování studenta při psaní SQL dotazu. U všech čtyř skupin se potvrdil sestupný trend chybovosti. Výsledky byly dále rozděleny podle použitého uživatelského rozhraní. U skupiny, jež měla nejlepší průměr chybovosti, je sestupný trend poměru chyb nejvýraznější, viz. Obrázek 19 Grafické zdůraznění trendu chybovosti v jednotlivých výkonnostních skupin. V této skupině bylo dosaženo největšího rozdílu mezi výchozí chybovostí a výslednou chybovostí. Zároveň tato skupina má největší rozpětí chybovosti.

Vstupní soubor obsahuje chybovost a četnost dotazů v rámci celé výuky seskupené podle jednotlivých minut, ve kterých docházelo ve třídě k aktivitě studentů.

Při rozdělení studentů do skupin podle jejich výkonnosti můžeme konstatovat, že čím větší

počet dotazů studenti při výuce procvičí, tím menší dosahují chybovosti na konci kurzu.

N_ERR_001; trans. = 0,3337+5,615E-7*x
N_ERR_002; trans. = 0,323+4,1622E-6*x
N_ERR_003; trans. = 0,3347-4,8159E-6*x
N_ERR_004; trans. = 0,3638-3,0031E-5*x



Obrázek 19 Grafické zdůraznění trendu chybovosti v jednotlivých výkonnostních skupin

5.2.3 Kategorizace chybovosti studentů

Studenty jsem rozdělil do skupin podle jejich průměrné chybovosti v jednotlivých lekcích do čtyř skupin, v intervalech daných vypočtenou chybovostí.

Do základních charakteristik chybovosti jednotlivých studentů byly zahrnuty průměrná chybovost, směrodatná odchylka chybovosti, rozptyl chybovosti, minimum, maximum. Chybovost byla vypočítána jako poměr správných odpovědí studenta v průběhu semestru vůči celkovému počtu odpovědí studenta.

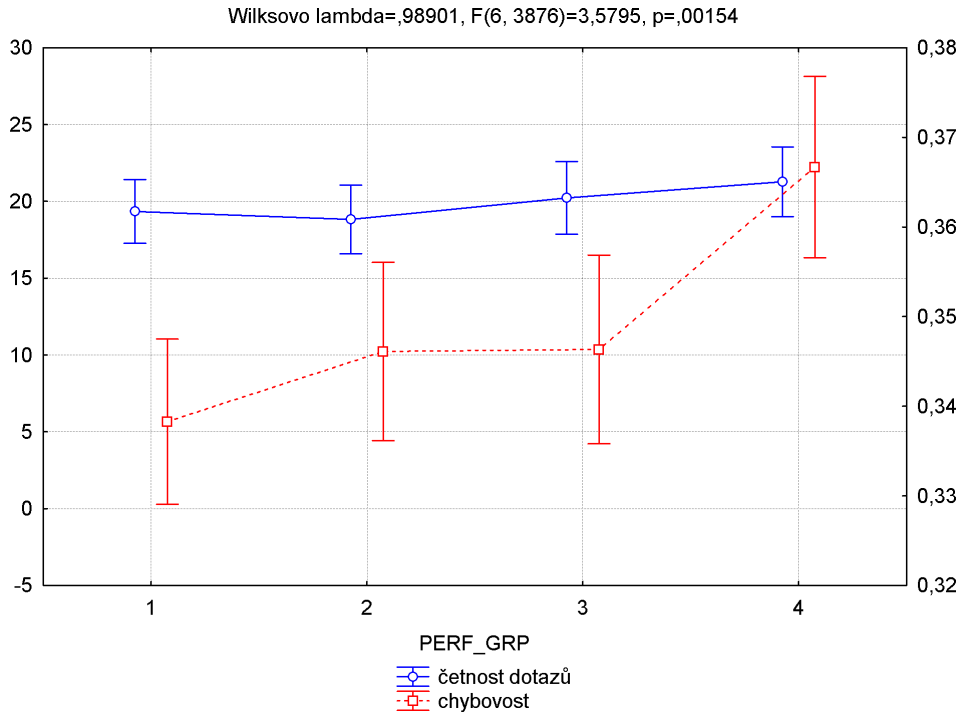
Student	Průměrná chybovost	Počet dotazů	Směrodatná odchylka	Rozptyl	Směrodatná chyba	Minimum	Maximum
1	0,35	103,00	0,12	0,01	0,01	0,10	0,80
2	0,34	81,00	0,10	0,01	0,01	0,11	0,67
3	0,35	63,00	0,10	0,01	0,01	0,14	0,67
4	0,35	64,00	0,12	0,01	0,01	0,19	0,75
5	0,37	65,00	0,13	0,02	0,02	0,13	0,67
6	0,34	89,00	0,11	0,01	0,01	0,13	0,67
7	0,36	55,00	0,12	0,01	0,02	0,17	0,67
8	0,38	74,00	0,11	0,01	0,01	0,17	0,67
9	0,38	105,00	0,13	0,02	0,01	0,15	0,88
10	0,35	81,00	0,11	0,01	0,01	0,13	0,75
11	0,36	82,00	0,13	0,02	0,01	0,13	0,75
14	0,30	67,00	0,10	0,01	0,01	0,10	0,75
15	0,34	88,00	0,11	0,01	0,01	0,10	0,67
16	0,35	62,00	0,11	0,01	0,01	0,10	0,67
17	0,35	72,00	0,10	0,01	0,01	0,13	0,67
18	0,34	61,00	0,11	0,01	0,01	0,10	0,71
19	0,34	95,00	0,12	0,02	0,01	0,10	0,80
20	0,35	98,00	0,11	0,01	0,01	0,08	0,67
21	0,34	72,00	0,11	0,01	0,01	0,06	0,55
22	0,33	118,00	0,09	0,01	0,01	0,17	0,67
23	0,36	51,00	0,11	0,01	0,02	0,13	0,67
24	0,36	100,00	0,11	0,01	0,01	0,15	0,75
25	0,36	83,00	0,12	0,01	0,01	0,17	0,71
26	0,34	61,00	0,11	0,01	0,01	0,13	0,67
28	0,30	53,00	0,08	0,01	0,01	0,10	0,50
Celkem	0,35	1943,00	0,11	0,01	0,00	0,06	0,88

Tabulka 13 Popisné statistiky chybovosti studentů

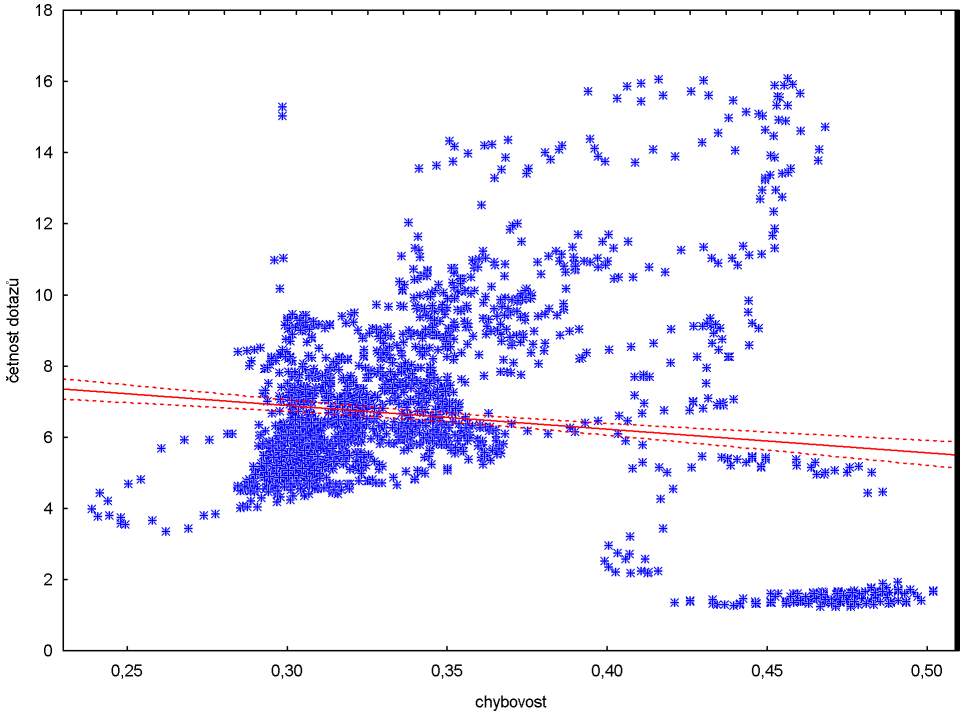
V rámci analýzy jsem jako první zkoumal rozdíl v chybovosti jednotlivých studentů. Studenti dosahují při učení databázových systémů statisticky významný rozdíl v chybovosti.

	SČ	SV	PČ	SČ	SV	PČ	F	p
Chybovost	0,57	24,00	0,02	23,74	1918,00	0,01	1,91	0,00

Tabulka 14 Analýza rozptylu chybovosti jednotlivých studentů



Obrázek 20 Analýza rozptylu pro chybovost a počet položených dotazů



Obrázek 21 Souvislost chybovosti a četnosti dotazů studentů

	chybovost Param.	chybovost Sm.Ch.	chybovost t	chybovost p	-95,00% LmtSpol.	+95,00% LmtSpol.	chybovost Beta (ß)	chybovost Sm.Ch. ß
Abs. člen	0,37	0,003	111	0	0,36	0,4		
četnost dotazů	-0,003	0,0005	-5,96	0	-0,0036	-0,0018	-0,14	0,02

Tabulka 15 Analýza rozptylu závislosti chybovosti a četnosti dotazů studentů

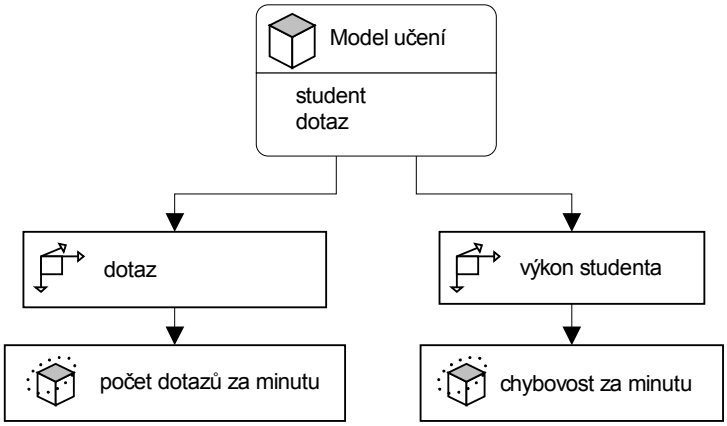
Při analýze chybovosti byla použita jednofaktorová analýza rozptylu. Jako závislá proměnná byla chybovost a nezávislá proměnná student.

Na základě výpočtu analýzy rozptylu jednotlivých studentů lze konstatovat, že existuje statisticky významný rozdíl ve výkonech jednotlivých studentů. Kromě toho lze na základě výsledků usuzovat na souvislost mezi počtem zadaných dotazů a chybovostí. Čím větší je počet zadaných dotazů, tím je menší chybovost studenta. Z toho vyplývá, že při učení dotazovacího jazyka dochází k učení z chyb.

5.3 Vztah chybovosti a četnosti dotazů

5.3.1 Metodika výpočtu

Východiskem koncepce metodiky výpočtu souvislosti mezi počtem dotazů zadaných studenty a jejich chybovostí je analýza vstupních proměnných a definice výzkumného pole. V prvním kroku stanovení analyzovaných metrik byla provedena konstrukce statistického souboru. Při ní byly vybrány proměnné student a dotaz. Konkrétně se jednalo o dimenzi dotaz a výkon studenta. Z modelu byly stanoveny metriky počet dotazů za minutu a chybovost za minutu.



Obrázek 22 Model výzkumného pole pro hypotézu

Chybovost studenta v rámci lekce byla ve všech výpočtech vyjádřena poměrem celkového počtu správných dotazů a celkového počtu dotazů v úhrnu. Pokud označíme $\sum_{i=1}^{n} x_i$ celkový počet chybných SQL dotazů a $\sum_{i=1}^{n} y_i$, můžeme tento poměr vyjádřit následující rovnicí:

$$\rho = \frac{\sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i}$$

kde n je počet lekcí absolvovaných studentem v rámci semestru databázových systémů. V čitateli je suma celkového počtu chyb studenta v jednotlivých lekcích, kterých se dopustil

v rámci semestru při zadávání příkazů v dotazovacím jazyce. Ve jmenovateli je součet celkového počtu správných a nesprávných odpovědí studenta, kterých se dopustil při zadávání dotazů v dotazovacím jazyce v rámci předmětu databázové systémy.

Před zahájením výpočtů byla provedena standardizace, ve které jsem z dat odstranil krajní hodnoty. Jednalo se o ty hodnoty, které byly menší než polovina průměru a větší než 50 % průměru. Jako první jsem vypočetl popisnou statistiku vstupních proměnných. Ve statistickém souboru se tyto nevalidní údaje vyskytovaly vlivem chyb při přihlašování studentů do databáze. Cílem bylo zjistit základní charakteristiky vstupního statistického souboru. Ten byl dále rozdělen na celkovou chybovost, chybovosti v textovém rozhraní a chybovosti v grafickém rozhraní.

	N platných	Průměr	Minimum	Maximum	Sm. odch.
celková chybovost	578	0,36	0,05	0,83	0,11
chybovost textové rozhraní	195	0,39	0,05	0,83	0,16
chybovost grafické rozhraní	383	0,33	0,13	0,71	0,07

Tabulka 16 Popisné statistiky chybovosti

V dalším kroku byly vytipovány pro analýzu souvislostí mezi počtem dotazů a chybovostí statistické metody korelace a lineární regrese. Vybrané statistické metody byly aplikovány a byla potvrzena jejich použitelnost.

Vzhledem k charakteru aplikovaných statistických metod bylo před výpočtem konkrétních hodnot nezbytné ověřit, zdali analyzované proměnné nabývají normálního rozdělení. Ověření proběhlo pomocí Kolmogorov-Smirnova Testu. Ten je založen na míře maximálního rozdílu mezi kumulativním rozdělením vstupní proměnné a hypotetickým kumulačním rozdělením. Druhým použitým testem normality byl Lilieforsův pravděpodobnostní test. Třetím použitým testem normality byl Shapiro-Wilkův W Test.

	N	max D	K-S p	Lilliefors p	W	p
celková chybovost	578	0,12	p < ,01	p < ,01	0,93	0
chybovost textové rozhraní	195	0,10	p < ,05	p < ,01	0,98	0,01
chybovost grafické rozhraní	383	0,10	p < ,01	p < ,01	0,94	0

Tabulka 17 Test normality chybovosti

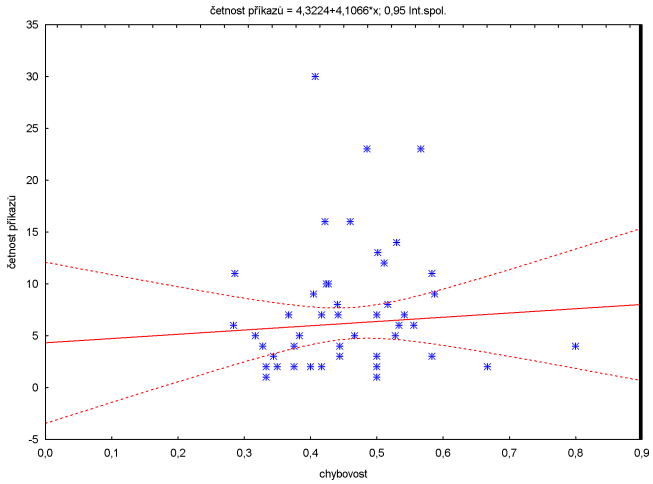
Z výsledku bylo patrné, že chybovost měla ve všech třech případech normální rozdělení

5.3.2 Korelace četnosti a chybovosti dotazů

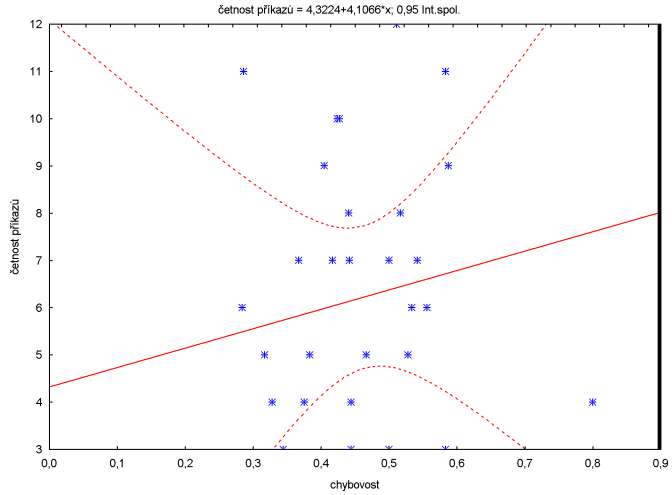
Korelace mezi četností a chybovostí byla rozdělena do tří skupin. První popisuje vztah mezi chybovostí a četností v textovém rozhraní, druhá v grafickém rozhraní a třetí korelaci celkovou.

	Korelace p < ,05000 N=195	
	Četnost	chybovost
Četnost	1,00	-0,18
chybovost	-0,18	1,00

Tabulka 18 Korelace proměnných četnost a chybovost pro textové rozhraní



Obrázek 23 Korelace chybovosti a četnosti dotazů v textovém rozhraní



Obrázek 24 Korelace chybovosti a četnosti dotazů v textovém rozhraní bez odlehlých hodnot

Vztah četnosti a chybovosti dotazů byl determinován negativní slabou korelací na hladině významnosti 0,05.

	Korelace	p < ,05000 N=383
	četnost	chybovost
Četnost	1,00	-0,03
chybovost	-0,03	1,00

Tabulka 19 Korelace proměnných četnost a chybovost pro grafické rozhraní

Korelace mezi proměnnými četnost a chybovost není statisticky významná na hladině významnosti 0,05.

	Korelace p < ,05000 N=578	
	Četnost	chybovost
Četnost	1,00	-0,14
chybovost	-0,14	1,00

Tabulka 20 Celková korelace proměnných četnost a chybovost

Korelace mezi proměnnými četnost a chybovost je statisticky významná na hladině významnosti 0,05.

5.3.3 Korelace

Pro kvantifikaci vztahu mezi počtem dotazů a chybovostí byl proveden výpočet korelačního koeficientu. Ten byl aplikován na statistický soubor proměnných X a Y, kde proměnná X vyjadřovala četnost dotazů studenta za minutu a proměnná Y chybovost studenta za minutu. Výsledek lze interpretovat jako kvantitativní vyjádření asociace obou proměnných. Pro výpočet korelačního koeficientu ρ_{XY} byl použit následující vzorec:

$$r \S r_{XY} = \frac{s_{XY}}{s_X s_Y}$$

kde s_{XY} je kovariace X a Y, která se vypočítá jako:

$$s_{SY} = \sum_{i=1}^{n} \frac{(x_i - \bar{x})(y_i - \bar{y})}{(n - 1)}$$

Při interpretaci korelačního koeficientu nalézajícího se v intervalu [-1,1] a vyjadřujícího stupeň asociace mezi sledovanými proměnnými byl použit následující klíč:
Pokud nabývá hodnoty 0: není lineární asociace (X a Y jsou lineárně nekorelované),
pokud nabývá hodnoty 1: úplná lineární asociace, kde X a Y se liší ve stejném směru,
pokud nabývá hodnoty -1: úplná lineární asociace, kde se X a Y liší v opačném směru.
Korelační koeficient výsledků byl uspořádán do symetrické korelační matice, kde každý prvek je korelačním koeficientem kombinace řádku a sloupce. (Hendl, 2004)

5.3.4 Regresní analýza chybovosti a četnosti dotazů

Pro podrobnější analýzu souvislosti mezi četností a chybovostí dotazů byla vybrána regresní analýza. Byla aplikována ve třech případech. V prvním případě regrese kvantifikovala vztah četnosti a chybovosti v textovém rozhraní, ve druhém grafického rozhraní a třetí celkem.

	Hodnota
Vícenás. R	0,18
Vícenás. R^2	0,03
Přizpůs. R^2	0,03
F(1,193)	6,59
P	0,01
Sm. chyba odhadu	0,16

Tabulka 21 Souhrn regrese chybovosti a četnosti dotazů v rámci textového rozhraní

	Beta	Sm.chyba beta	B	Sm.chyba B	t(193)	Úroveň p
Abs.člen			0,43	0,018	24,13	0
četnost	-0,18	0,07	-0,0034	0,0013	-2,57	0,01

Tabulka 22 Souhrn regrese chybovosti a četnosti dotazů v textovém rozhraní

Výsledek regrese ukazuje, že četnost dotazů v textovém rozhraní je statisticky významnou vysvětlující proměnnou pro chybovost v textovém rozhraní. Vztah ukazuje, že čím menší je četnost dotazů, tím větší je chybovost.

	Hodnota
Vícenás. R	0,03
Vícenás. R^2	0,001
Přizpůs. R^2	-0,002
F(1,381)	0,41
P	0,52
Sm. chyba odhadu	0,068

Tabulka 23 Souhrn regrese chybovosti a četnosti dotazů v rámci grafického rozhraní

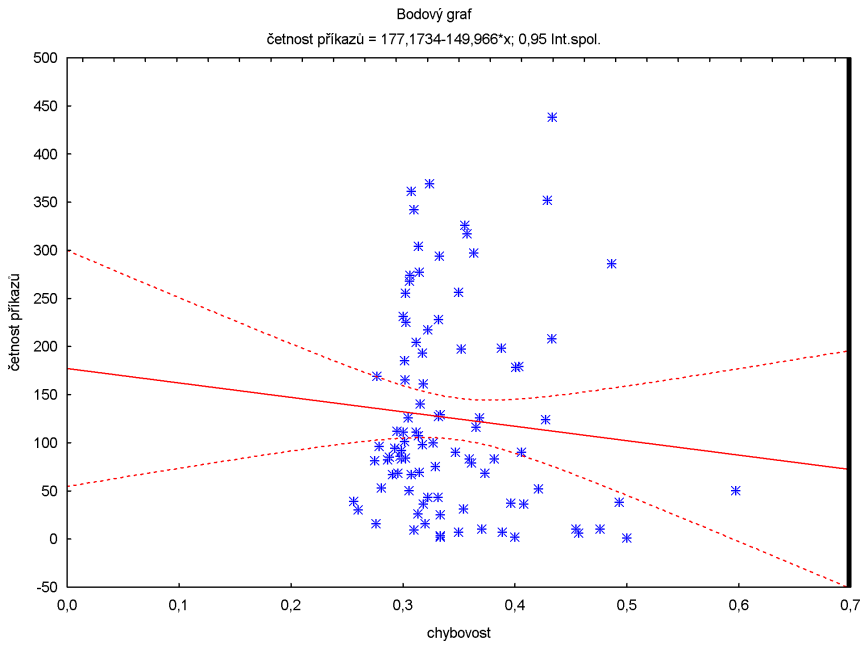
	Beta	Sm.chyba beta	B	Sm.chyba B	t(381)	Úroveň p
Abs.člen			0,34	0,0048	70,33	0
Četnost	-0,033	0,05	-0,000021	0,000033	-0,64	0,52

Tabulka 24 Souhrn regrese chybovosti a četnosti dotazů v rámci grafického rozhraní

Mezi chybovostí studentů a počtem dotazů při učení databázových systémů v rámci semestru existuje statisticky významný vztah. Hypotéza je omezená pouze na grafickém rozhraní.

	Odhady parametrů					
	chybovost Param.	Chybovost Sm.Ch.	chybovost t	chybovost p	-95,00% LmtSpol.	+95,00% LmtSpol.
Abs. člen	0,35	0,00995	35,47	0	0,33	0,37
četnost příkazů	-0,000052	0,000061	-0,86	0,34	-0,00017	0,000069

Tabulka 25 Výsledek jednoduché lineární regrese



Tabulka 26 Graf souvislosti četnosti příkazů s chybovostí

Výsledek regrese neukázal statisticky významné vysvětlení chybovosti četností dotazů v rámci grafického rozhraní.

Mezi chybovostí studentů a počtem dotazů při učení databázových systémů v rámci semestru existuje statisticky významný vztah. Hypotéza je omezená pouze na textové rozhraní.

	Chybovost Param.	Chybovost Sm.Ch.	Chybovost t	Chybovost p	Chybovost Beta (ß)	Chybovost Sm.Ch. ß
Abs. člen	0,45	0,02	26,17	0		
četnost příkazů	0,001	0,002	0,5	0,6	0,06	0,13

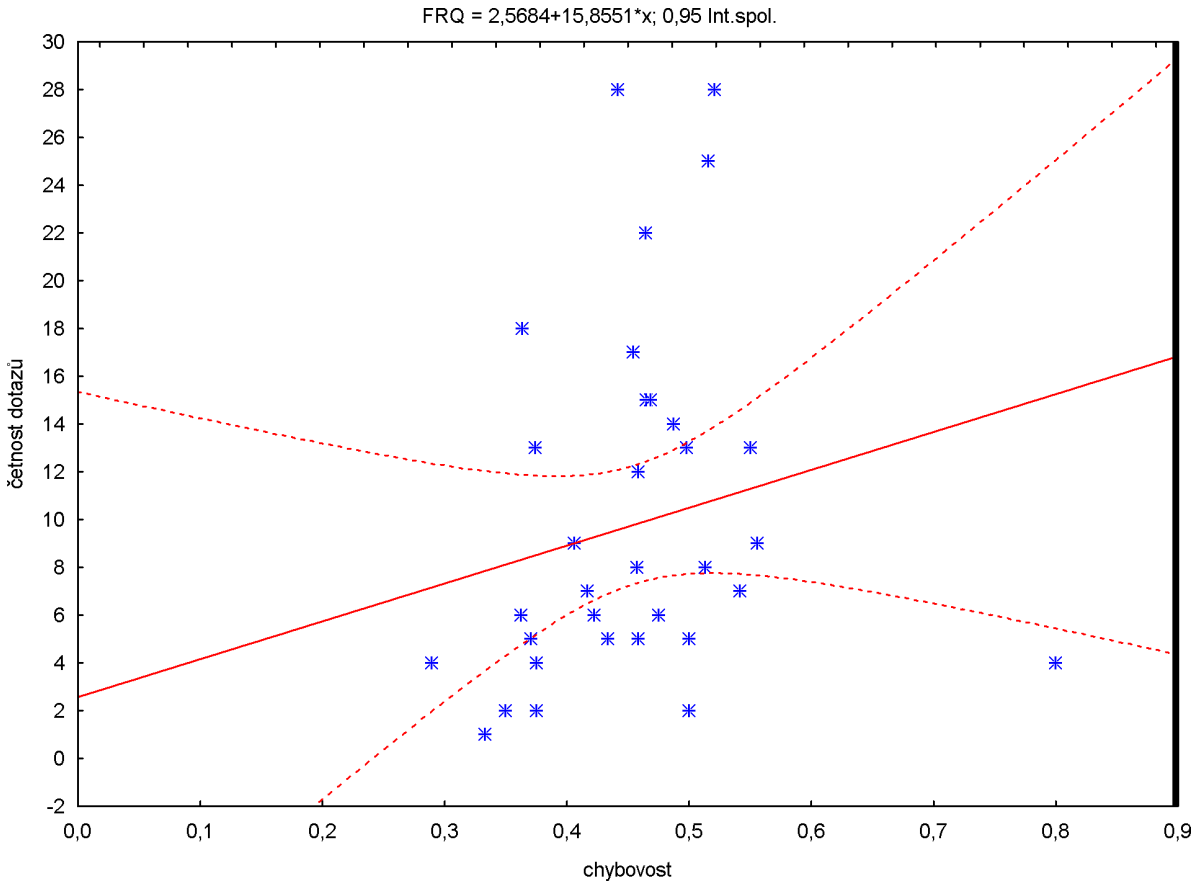
Tabulka 27 Regrese chybovosti a četnosti dotazů v rámci textového rozhraní

	Hodnota
Vícenás. R	0,14
Vícenás. R^2	0,02
Přizpůs. R^2	0,02
F(1,576)	11,53
P	0,00073
Sm. chyba odhadu	0,1

Tabulka 28 Souhrn regrese pro celkovou chybovost a četnost dotazů

C	Beta	Sm.chyba beta	B	Sm.chyba B	t(576)	Úroveň p
Abs.člen			0,367	0,0058	63,7	0
Četnost	-0,14	0,04	-0,000167	0,000049	-3,4	0,0007

Tabulka 29 Souhrn regrese pro celkovou chybovost a četnost dotazů



Obrázek 25 Korelační diagram chybovosti a četnosti dotazů

Výsledný regresní koeficient byl -0,0028. Lze jej interpretovat tak, že na každou jednotku snížení dotazu připadá snížení chybovosti o 0,0028 na statistické hladině významnosti $p<0.05$. Koeficient Beta určuje, že pro každé snížení standardní odchylky četnosti dotazů je 0,14 snížení odchylky v rámci četnosti dotazů.

Výsledek regrese ukazuje, že četnost dotazů je statisticky významnou vysvětlující proměnnou pro celkovou chybovost. Výsledky je možné interpretovat tak, že je čím menší četnost dotazů, tím větší je chybovost.

5.3.5 Lineární regrese

Druhou metodou, jež byla aplikována v rámci analýzy vztahu počtu dotazů a chybovosti byla jednoduchá lineární regrese. Cílem aplikace regresní analýzy bylo ověření potenciálu jiného způsobu kvantifikace rozvoje databázového myšlení. Jako determinující proměnná byl vybrán počet dotazů. Jako závislá proměnná byla použita chybovost studenta. Výběr metody a proměnných poskytuje možnost sledovat, do jaké míry má aktivita studenta vliv na zlepšení

databázového myšlení. Při výpočtu jednoduché lineární regrese byl aplikován následující regresní model:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

kde

a) Y_i je náhodná proměnná reprezentující sledovanou proměnnou y_i pro každou předpovídanou proměnnou x_i. V tomto případě se jedná o počet dotazů studenta za minutu. Y_i je distribuována jako $f_{Y_i}(y)$. Lineární regresní parametry, β_0 a β_1 jsou známy jako hranice a sklon.

b) ε_i je proměnná chyba měření s:

$$E[\varepsilon_i] = 0; V[\varepsilon_i] = \sigma^2; V[\varepsilon_i \varepsilon_j] = 0; \forall i \neq j.$$

Chyby měření jsou považovány za nulové, stejného rozptylu a nejsou mezi sebou korelované. S těmito předpoklady lze derivovat následující model:

a) chyby s

$$E[\varepsilon_i] = 0 \Rightarrow E[Y_i] = \beta_0 + \beta_1 x_i \Rightarrow E[Y] = \beta_0 + \beta_1 X.$$

poslední rovnice vyjadřuje lineární závislost Y na X. Lineární regresní parametry β_0 a β_1 mohou být odhadnuty z dat. Hustota sledovaných proměnných $f_{Y_i}(y)$ je hustota chyb $f_\varepsilon(\varepsilon)$ s překladem průměru na $E[Y_i]$.

b)

$$V[\varepsilon_i] = \sigma^2 \Rightarrow V[Y_i] = \sigma^2$$

c) tam, kde Y_i a Y_j jsou nekorelované.

V odhadu regresní funkce je aplikována metoda nejmenších čtverců. Její princip je založen na minimalizaci celkových součtů čtverců chyb (odchylek) mezi sledovanými proměnnými y_i

a odhadovanými hodnotami $b_0 + b_1 x_i$

$$E = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - b_0 - b_1 x_i)^2$$

kde b_0 a b_1 jsou odhady β_0 a β_1

K tomu, aby bylo možné aplikovat metodu nejmenších čtverců, je nutné provést diferenciaci E po b_0 a b_1 a vyrovnat jej nule, čímž lze získat tzv. normalizovanou rovnici:

$$\begin{aligned} \sum y_i &= nb_0 + b_1 \sum x_i \\ \sum x_i y_i &= b_0 \sum x_i + b_1 \sum x_i^2 \end{aligned}$$

kde součet je vždycky počítán jako n hodnot prediktivních. Tím, že se vyřeší normální rovnice, následuje odhad parametru b_0 a b_1, ze kterých je získáno:

$$b_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$$
$$b_0 = \bar{y} - b_1 \bar{x}$$

Mezi charakteristiky odhadu nejmenších čtverců, jež je nutné brát v patrnost, patří:

a) parametry b_0 a b_1 jsou nevyvážené odhady skutečného parametru β_0 a β_1 $(E[b_0] = \beta_0, E[b_1] = \beta_1)$, kde máme minimální rozptyl v rámci všech nevyvážených lineárních odhadů.

b) předpovídané hodnoty $\hat{y} = b_0 + b_1 x_i$ jsou bodové odhady pravdivých, sledovaných proměnných y_i. Stejně správné je celá rovnice $Y = b_0 + b_1 X$, kde je bodový odhad průměrný $E[Y]$,

c) regresní přímka vždy prochází body $(\bar{x}, \bar{y})$,

d) výpočet chyb $e_i = y_i - \hat{y}_i = y_i - b_0 - b_1 x_i$ se jmenuje rezidua a jedná se o bodový odhad chybových hodnot ε_i. Součet všech reziduí je nula: $\sum e_i = 0$.

e) rezidua jsou nekorelovaná s prediktorem a předpovídanými hodnotami:

$\sum e_i x_i = 0; \sum e_i \hat{y}_i = 0$

f) $\sum y_i = \sum \hat{y}$ a $\bar{y} = \bar{\hat{y}}$, když například předpovídané hodnoty mají stejné průměry jako sledované hodnoty.

Kromě metody nejmenších čtverců lze při výpočtu lineární regrese použít alternativně i jiné míry chyb. Minimalizace čtverců chyb tak může být vyjádřena jako minimalizovaný součet absolutních hodnot chyb $E = \Sigma |\varepsilon_i|$.

V rámci výsledků je hodnota Beta, spojena s tzv. standardizovaným regresním modelem:

$$Y_i^* = \beta_1^* x_i^* + \varepsilon_i$$

V rámci rovnice může být jeden parametr použit, protože Y_i^* a x_i^* jsou standardizované proměnné (průměr = 0, standardní odchylka = 1) sledovaných a předpovídaných proměnných. Může být dokázáno, že:

$$\beta_i = \left(\frac{\sigma_Y}{s_x} \right) \beta_1^*$$

kde standardizované β_1^* je tzv. beta koeficient, který má bodový odhad právě výsledek v kolonce Beta.

Při výpočtu lineární regrese docházíme kromě jiného i k vyjádření koeficientů R, R^2 a upraveného R^2. Nejedná se však o míry asociace k posouzení přesnosti pasování modelu. Při jejich výpočtu byl nejdříve vypočítán součet chyb čtverců (SSE):

$$SSE = \sum (y_i - \hat{y}_i)^2 = \sum e_i^2$$

Vzhledem k tomu, že byly ztraceny dva stupně volnosti b_0 a b_1, musí mít SSE $n - 2$ stupňů. Na základě toho vypočteme průměr čtverce chyb:

$$MSE = \frac{SSE}{n - 2}$$

Celková variace sledovaných proměnných vypočtena jako Total Sum of Squares (SST):

$$\text{SST} = \text{SSY} = \sum (\text{y}_\text{i} - \bar{y})^2$$

Dalším vyjádřením míry asociace je koeficient determinace nebo R čtverec:

$$r^2 = \frac{SST - SSE}{SST} \in [0,1]$$

proto je „R-čtverec“, neboli čtverec Pearsonovy korelace mezi x_i a y_i, měřen přispěním X a redukcí rozptylu Y. Tedy v redukci nekonkrétnosti v předpovědi Y. (Hendl, 2004)

5.4 Vliv vstupních předpokladů

5.4.1 Metodika výpočtu

Cílem kapitoly bylo vybrat vhodnou metodu pro analýzu souvislosti vstupních předpokladů studenta a kvantitativní charakteristiky rozvoje jeho databázového myšlení. V rámci stanovení vstupních předpokladů, jež budou zahrnuty do analýzy, byly ze vstupního dotazníku vybrány dvě otázky vhodné pro zpracování. Konkrétně se jednalo o otázku týkající se zkušeností se vzdělávacím softwarem a otázku ohledně původu znalostí v oblasti počítačové gramotnosti. U jednotlivých otázek byl analyzován jejich vliv na chybovost studenta při učení dotazovacího jazyka. Kromě toho byl vliv zkoumán odděleně pro učení v grafickém a textovém rozhraní. Při koncepci vhodné struktury zdrojového statistického souboru byly zahrnuty proměnné odpověď na vstupní otázku, student, čas a výsledek učení.

Pro zjištění vlivu výše uvedených faktorů na závislou proměnnou byla jako relevantní identifikována vícenásobná analýza rozptylu. Vícenásobná analýza rozptylu (Analysis of Variance) je určena k analýze simultánního vlivu více nezávislých proměnných. Základní statistikou v analýze rozptylu je F-testovací statistika rozdílnosti skupinových průměrů, pomocí níž se testuje hypotéza, zda průměry ve skupinách určených kombinacemi faktorů se od sebe liší více než na základě působení náhodného kolísání. Pokud se průměry neliší významně, usuzujeme, že faktory nemají na závisle proměnnou vliv. Jestliže F-test indikuje nějaký systematický vliv, používají se testy simultánního srovnávání pro nalezení kombinací hodnot faktorů, které nejvíce přispívají k systematickým vlivům. (Hendl, 2004)

Při stanovování metodiky výpočtu faktorové analýzy byly v prvním kroku vybrány nezávislé proměnné. Jako nezávislé proměnné byly vybrány četnosti odpovědí na otázky ze vstupního dotazníku. Za závislou proměnnou byla vybrána chybovost studenta při učení dotazovacího jazyka. Ze vstupního dotazníku byly do analýzy rozptylu zahrnuty dvě otázky. Jako první byla vybrána otázka „Kdo Vás naučil nejvíce z toho, co víte o počítačích a jejich používání?“

V dalším kroku byl proveden výpočet četností na jednotlivé otázky. Nejčastěji se studenti kladně odpovídali u odpovědi „Vysoká škola“, druhá nejčastější odpověď byla „Moji přátelé“ a třetí odpověď „Naučil jsem se sám“.

Ot. č. 6	Kdo Vás naučil nejvíc z toho, co víte o počítačích a jejich používání?	četnost
a)	Základní škola	0/16
b)	Střední škola	3/16
c)	Vysoká škola	9/16
d)	Moji přátelé	8/16
e)	Moje rodina	2/16
f)	Naučil jsem se sám	8/16
g)	Jinde	4/16

Tabulka 30 Četnosti odpovědí otázky č. 6

Jako druhá byla ze vstupního dotazníku vybrána otázka „Počítač používám nejčastěji pro“. Nejčetnější odpověď na otázku byla odpověď „Internet pro elektronickou komunikaci“, druhá nejčetnější odpověď byla „Psaní textových dokumentů“ a třetí nejčetnější odpověď byla „Kreslení, malování nebo používání grafických programů“.

Ot. č. 3	Počítač používám především pro:	Odpověď Ano
a)	Vyhledávání lidí, věcí a myšlenek na internetu	12/16
b)	Hraní počítačových her	8/16
c)	Psaní textových dokumentů	15/16
d)	Spolupráci s lidmi a týmy po internetu	6/16
e)	Práci s tabulkovým procesorem	8/16
f)	Stahování softwaru (i her)	10/16
g)	Kreslení, malování nebo používání grafických programů	13/16
h)	Práci se vzdělávacím softwarem	8/16
i)	Internet pro stahování muziky	8/16
j)	Internet pro elektronickou komunikaci (např.: e-mail, chat)	16/16

Tabulka 31 Četnosti kladných odpovědí u otázky „Počítač používám především pro ...“

Výpočet analýzy rozptylu vychází z předpokladu, že některý z faktorů bude mít vliv na výsledek učení studenta. V rámci první fáze analýzy jsem na základě výsledků analýzy rozptylu u jednotlivých položek dotazníku na hladině významnosti 0,05 identifikoval jako statisticky významné dva faktory. První z nich byl vliv zkušenosti se vzdělávacím softwarem. Druhý faktor byl vliv zdroje poznatků o počítači od rodiny. V rámci výzkumu jsem sledoval působení těchto nezávislých proměnných na závislou proměnnou chybovosti studentů při rozvoji databázového myšlení v podobě učení dotazovacího jazyka.
Pro podrobnější analýzu jsem použil analýzu rozptylu, ve které jsem v prvním případě sledoval souvislost mezi:

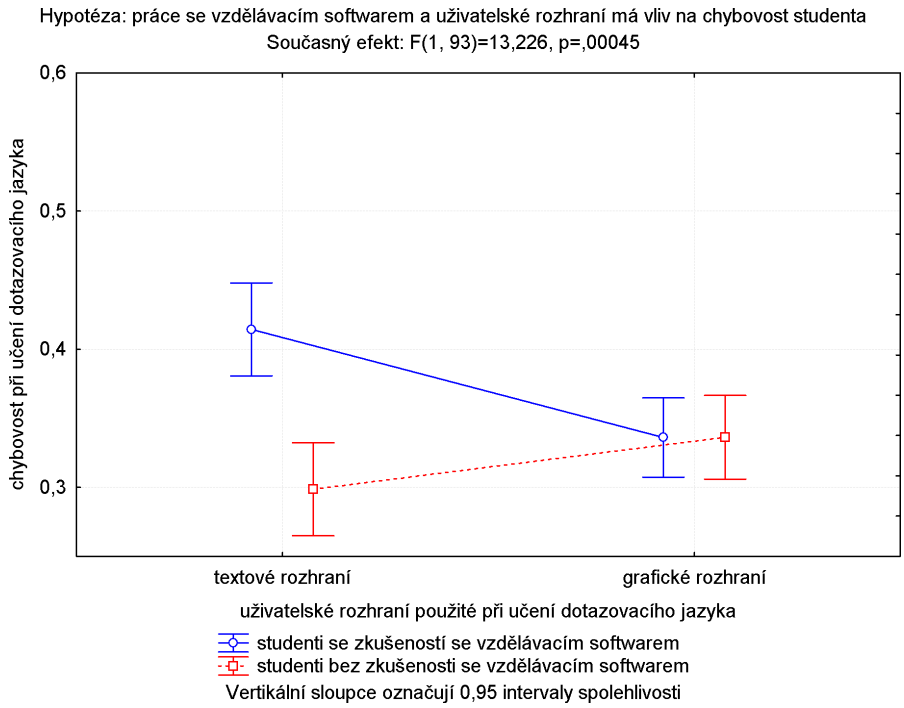
1) Zkušenostmi se vzdělávacím softwarem.

2) Učebním výsledkem vyjádřeným chybovostí.
3) Použitím uživatelského rozhraní při učení.
4) Rodinou jako zdrojem znalostí o počítačích.
5) Učebním výsledkem vyjádřeným chybovostí.
6) Použitým uživatelským rozhraním při učení.

Při stanovování metodiky výpočtu faktorové analýzy byla pro každou otázku provedena faktorová analýza rozptylu odděleně. Jako druhý faktor byl stanoven druh použitého prostředku při učení dotazovacího jazyka. Dále byla analýza rozptylu rozdělena na textové rozhraní a grafické rozhraní. Výsledky lze interpretovat tak, že zkušenost se vzdělávacím softwarem má statisticky významný vliv na chybovost při učení dotazovacího jazyka v textovém uživatelském rozhraní. Kromě toho zkušenost se vzdělávacím softwarem má statisticky významný vliv na chybovost při učení dotazovacího jazyka v grafickém uživatelském rozhraní.

zkušenosti se vzdělávacím softwarem	uživatelské rozhraní	chybovost průměr	chybovost sm. ch.	chybovost - 0,95	chybovost 0,95	N
Ano	textové	0,41	0,02	0,38	0,45	21,00
Ano	grafické	0,34	0,01	0,31	0,36	29,00
Ne	textové	0,30	0,02	0,27	0,33	21,00
Ne	grafické	0,34	0,02	0,31	0,37	26,00

Tabulka 32 Vícefaktorová analýza rozptylu učebního výsledku



Obrázek 26 ANOVA vzdělávací software a uživatelské rozhraní

Na základě výsledků faktorové analýzy rozptylu konstatuji, že zkušenost se vzdělávacím softwarem má statisticky významný vliv na chybovost při učení dotazovacího jazyka v textovém uživatelském rozhraní.

Na základě výsledků faktorové analýzy rozptylu konstatuji, že zkušenost se vzdělávacím softwarem má statisticky významný vliv na chybovost při učení dotazovacího jazyka v grafickém uživatelském rozhraní.

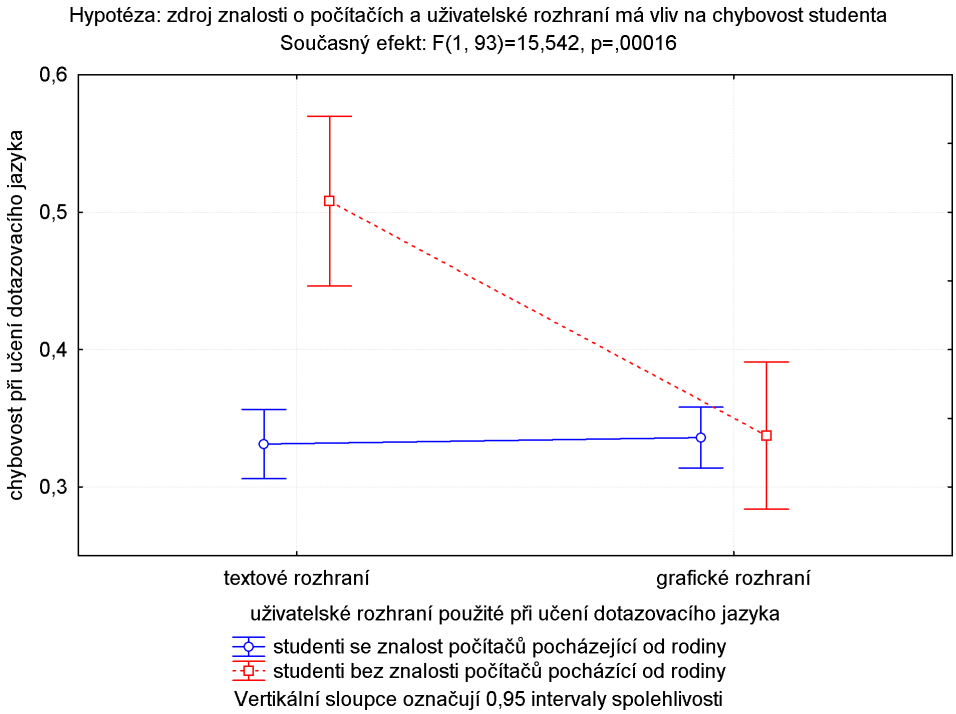
Další hypotéza může porovnávat oba způsoby učení a na jejich základě předpokládat, že studenti se zkušeností se vzdělávacím softwarem dosahují větší chybovosti při použití textového rozhraní. Oproti studentům, kteří zkušenosti se vzdělávacím softwarem nemají, dosahují menší chybovosti při práci s textovým rozhraním. V rámci grafického rozhraní tento efekt nelze sledovat.

Výsledky lze interpretovat tak, že rodina jako zdroj informací o počítačích má statisticky významný vliv na chybovost při učení dotazovacího jazyka v textovém uživatelském rozhraní. Dále rodina jako zdroj informací o počítačích má statisticky významný vliv na chybovost při učení dotazovacího jazyka v grafickém uživatelském rozhraní. Pro ověření hypotézy byla vybrána metoda faktorové analýzy rozptylu.

počítačové znalosti od rodiny	Uživatelské rozhraní	chybovost Průměr	chybovost sm. ch.	chybovost -95%	chybovost +95%	N
ne	textové	0,33	0,01	0,31	0,36	36,00
ne	grafické	0,34	0,01	0,31	0,36	47,00
ano	textové	0,51	0,03	0,45	0,57	6,00
ano	grafické	0,34	0,03	0,28	0,39	8,00

Tabulka 33 Faktorová analýza rozptylu pro původ počítačových znalostí z rodiny

Na základě výsledků lze konstatovat, že počítačová znalost získaná od rodiny má statisticky významný vliv na chybovost při učení dotazovacího jazyka v grafickém uživatelském rozhraní. Počítačová znalost získaná od rodiny má statisticky významný vliv na chybovost při učení dotazovacího jazyka v textovém uživatelském rozhraní.



Obrázek 27 Výsledek faktorové analýzy rozptylu učebního výsledku

Ze zkoumání výsledků analýzy lze usuzovat, že studenti, kteří získali počítačové znalosti od své rodiny, dosahují stabilní chybovosti jak v textovém, tak i grafickém rozhraní. Oproti tomu studenti, kteří nemají počítačové znalosti od své rodiny, dosahují výrazně vyšší chybovosti při použití textového rozhraní a naopak výrazně nižší zkušenosti v grafickém rozhraní.

5.4.2 Analýza rozptylu

Pro analýzu vlivu vstupních předpokladů byla použita analýza rozptylu (ANOVA). Metoda ANOVA byla vybrána, neboť dokáže vyjádřit, jaké jsou statisticky významné rozdíly mezi skupinami. Obecně lze porovnat průměr dvou populací pomocí t-testu. Studie se ovšem neomezuje na porovnání pouze dvou skupin. Aby bylo možné porovnat libovolný počet průměrů, použiji analýzu rozptylu.

Při dvojném třídění zkoumáme vliv dvou faktorů na závislou proměnnou. Označme tyto faktory A a B. Vliv rušivého faktoru se snažíme oddělit od vlivu faktoru A.

Označme *a* počet úrovní faktoru *A* a *b* počet úrovní faktoru B. Počet objektů odpovídajících i-té úrovni faktoru A a j-té úrovni faktoru B označme n_{ij}. Nejčastěji se setkáváme s tzv. náhodnými úplnými bloky neboli vyváženým tříděním, kdy všechny četnosti n_{ij} jsou shodné. Efekty obou faktoru považujeme konstantní. Abychom mohli otestovat interakce mezi faktory, je vhodné, aby n_{ij} byly větší než 1. Pro naměřené metody uvažujeme model:

$$x_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \varepsilon_{ijk}$$

kde μ je společná část průměru závislé proměnné

α_i je efekt faktoru A na úrovni *i* (*i*=1,...,a),

β_j je efekt faktoru B na úrovni j (j=1,...,b),

γ_{ij} je interakce mezi faktorem A na úrovni *i* a faktorem *B* na úrovni *j*.

Poslední člen v rovnici označuje náhodnou chybu, o které předpokládáme, že má nulovou střední hodnotu, normální rozdělení a stejný rozptyl pro všechny hodnoty *i* a *j*, přičemž musí všechna měření představovat statisticky nezávislé náhodné proměnné. Pro každou kombinaci faktorů měříme objekt *c* objektů (k=1,....,c), platí tedy, že všechny $n_{ij} = c$. Předpokládáme, že hodnota *c* je větší než 1.

Při analýze zkoumáme tři páry hypotéz:

a) $H_{01}: \alpha_1 = \alpha_2 = \ldots = \alpha_a = 0.$
$H_{11}: Ne\,všechny\,efekty\,\alpha_i\,jsou\,nulové$

b) $H_{02}: \beta_1 = \beta_2 = \ldots = \beta_b = 0.$
$H_{12}: Ne\,všechny\,efekty\,\beta_j\,jsou\,nulové$

c) $H_{03}: Mezi\,faktory\,A\,a\,B\,není\,interakce\,(všechny\,\gamma_{ij} = 0).$
$H_{13}: Některé\,interakce\,jsou\,nenulové.$

Konstrukce testovací *F*-statistiky opět vychází z rozkladu součtu čtverců odchylek měření od společného průměru $\bar{x}$. Podobně jako u jednoduché analýzy rozptylu platí rozklad odchylky $(x_{ijk} - \bar{x})$ měření x_{ijk} od celkového průměru $\bar{x}$.

$$(x_{ijk} - \bar{x}) = (x_{ijk} - \bar{x}_{ij.}) + (\bar{x}_{i..} - \bar{x}) + (\bar{x}_{.j.} - \bar{x}) + (\bar{x}_{ij.} - \bar{x}_{i..} - \bar{x}_{.j.} + \bar{x})$$

Sčítanci $(\bar{x}_{i..} - \bar{x}), (\bar{x}_{.j.} - \bar{x}), (\bar{x}_{ij.} - \bar{x}_{i..} - \bar{x}_{.j.} + \bar{x})$ v tomto výrazu představuje odhady efektů α_i, β_j a γ_{ij}.

Jestliže sečteme čtverce obou stran rovnice pro všechna měření, dostaneme po úpravách:

$$S_T = \sum_i \sum_j \sum_k (x_{ijk} + \bar{x})^2$$
$$= \sum_i \sum_j \sum_k (x_{ijk} + \bar{x}_{ij.})^2 + ac \sum_i (\bar{x}_{i..} - \bar{x})^2 + bc \sum_j (\bar{x}_{.j.} - \bar{x})^2 + c \sum_i \sum_j (\bar{x}_{ij.} - \bar{x}_{i..} - \bar{x}_{.j.} + \bar{x})^2$$

Symbolicky tento výraz zapisujeme jako součet jednotlivých částí celkové variability:

$$S_T = S_e + S_A + S_B + S_I$$

Součty čtverců S_A a S_B zachycují hlavní efekty faktorů, součet S_I jejich interakce. Součet čtverců S_E hodnotí variabilitu měření uvnitř skupiny a slouží pro odhad společného rozptylu náhodné chyby ε_{ijk} v modelu analýzy rozptylu. Abychom mohli oprávněně prohlásit, že faktor *A* ovlivňuje cílovou proměnnou *X*, musí nutně platit nulová hypotéza H_{03}. Ta zaručuje, že

efekt faktoru *A* na dané úrovni je stejný pro všechny úrovně faktoru *B*.

5.5 Smíšené strategie analýzy výuky

Hlavním cílem disertační práce bylo přispět k rozvoji didaktiky informačních a komunikačních technologií. Ve stanoveném rámci navrhuji tedy metodiku sběru dat a směr komplexní analýzy dat ve výuce databázových systémů.

Metodika sběru dat by měla probíhat v několika oddělených fázích a současně v paralelních úrovních:

1) Vstupní fáze
    a. Vstupní dotazník
2) Hlavní fáze
    a. Zvukový význam vyučovací hodiny
    b. Záznam práce s výukovými materiály
    c. Záznam práce s databázovým systémem
    d. Záznam prezentovaných materiálů
3) Seminární fáze
    a. Sběr konceptuálních modelů
4) Výstupní fáze
    a. Didaktický test

Při výše popsaném sběru údajů by potom mohla komplexní analýza vstupních dat vypadat například následovně:

1) Komplexního zpracování zpracování záznamu vyučovací hodiny

Databázové a informační systémy I.			
Vyučující: Ing. Mikoláš Panský, 1.5 roku praxe			
Nahrál a protokol sestavil: Ing. Mikoláš Panský			
(vybraných 15 minut)			
začátek v souboru nahrávka014.amr od místa 46:10			
	U:	Kolik máme hodin? Jedenáct třicet tři, jsme tady do 12:15,	
		takže jsme přibližně někde v půlce. No a je tedy čas na to,	
		abysme se podívali na spojování tabulek. To bych už	
		více pojal jako cvičení. A proto ... N	
5	U:		hledá prezentaci
			v počítači
	U:	Ano, zde. Spojování tabulek. A proto začněme se spuštěním	
		MySQL Browser. MySQL Query Browser. Zase tedy připomínám:	
		Pokud nemáte … tady někde byl soubor, chytrý… někde na očku	
10		ve společném … N … Tak. Tady máte tedy buď tedy zadejte nové	
		spojení nebo se připojte pomocí tohoto druhého odstavce	
	U:		promítá na plátno
			obsah souboru,
			ve kterém jsou
15			instrukce, jak
			se připojit k serveru
	U:	Tak to už je rutina úplná	
	ŽŽ:		smích
			cvak, bum
20	U:	Tak .. Měli byste tam … chvilku počkáme… teď zase	
		uživatelské jméno zadejte vaše příjmení heslo zadejte	
		jméno vaše příjmení. Vaše jméno tečka vaše příjmení	
		Už to vypadá, že docela hodně lidí už tam je.	
		Mám takový pocit, že už všichni.	
25		Tak a zatím si zkusíme … Já doufám, že jsem ty práva nastavil	
		správně… Já se také přihlásím…	
	U:		ťuká do klávesnice,
			přilhašuje se
		atd.	

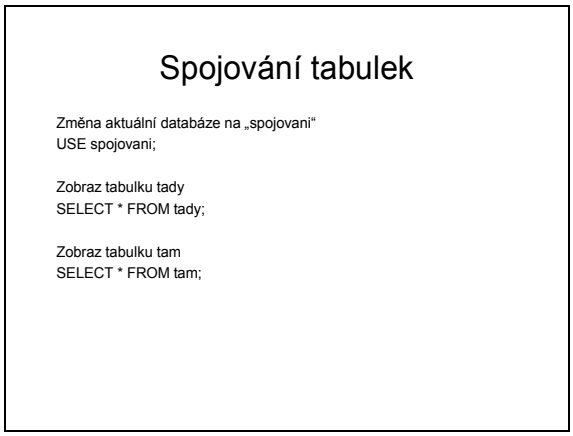
2) Záznam činnost studentů na výukovém databázovém serveru

Čas záznamu	Uživatel	Typ příkazu	Příkaz
2006-04-27 11:28:01	student014	Connect	student014@computer03 on
2006-04-27 11:28:01	student014	Query	SET SESSION interactive_timeout=1000000
2006-04-27 11:28:01	student014	Query	SELECT @@sql_mode
2006-04-27 11:28:01	student014	Query	SET SESSION sql_mode=''
2006-04-27 11:28:01	student014	Query	SET NAMES utf8
2006-04-27 11:28:01	student014	Init DB	student014
2006-04-27 11:28:01	student014	Init DB	student014
2006-04-27 11:28:01	student014	Query	show databases
2006-04-27 11:28:01	student014	Init DB	student014
2006-04-27 11:28:01	student014	Query	SHOW FULL TABLES
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table01`
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table02`
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table03`
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table04`
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table05`
2006-04-27 11:28:01	student014	Query	SHOW COLUMNS FROM `table06`
2006-04-27 11:28:01	student014	Init DB	student014
2006-04-27 11:28:01	student014	Init DB	student014
2006-04-27 11:28:01	student014	Query	SHOW PROCEDURE STATUS
2006-04-27 11:28:01	student014	Query	SHOW FUNCTION STATUS
2006-04-27 11:28:01	student014	Init DB	student014

3) Záznam z přístupu na materiály pro podporu výuky na výukovém serveru

K43	Thu 27. April 2006, 11:33	Počítač 3	Student014	resource view	Case Studio - relace
K43	Thu 27. April 2006, 11:33	Počítač 3	Student014	resource view	9. hodina
K43	Thu 27. April 2006, 11:33	Počítač 3	Student014	resource view	8. hodina

4) Ukázka z prezentace promítané při výuce



Výše uvedená ukázka nastiňuje možnost použití kvalitativních metod na základě nasbíraných údajů ve výuce. Jinou možností by bylo použít smíšenou strategii výzkumu, ve které by se zkombinovalo použití kvalitativních a kvantitativních metod.

6 Výsledky a interpretace

6.1 Diskuze

Stanovení rozsahu studie by mohlo být chápáno jako širší, než je obvyklé. Je tomu tak proto, že databázové systémy jsou velice rozsáhlou tématikou a existuje úzké propojení mezi jednotlivými podtématy mezi sebou. Již v praktické části bylo upuštěno od komplexní analýzy všech zaznamenaných materiálů a vymezil jsem se pouze na analýzu záznamů databázového serveru. Již ve fázi sběru dat jsem narazil na vysoký počet technických úskalí, které bylo nutné překonat. Oproti tomu byl rozsah zkoumaných materiálů rozšířen na vstupní dotazník, který mi umožnil komplexnější pohled na souvislost mezi vstupními předpoklady studentů a jejich výkonem v rámci vyučování. Pokud bych ale mohl experiment opakovat, bylo by v optimálním případě vhodné soustředit se především na detailní analýzu výkonů studenta z hlediska chybovosti. V souvislosti s tím je důležitá teorie ohledně křivek učení, jenž s tématem souvisejí. Díky širšímu rozsahu bylo však snazší naplnit cíl studie hledání metod sběru a analýzy dat ve výuce databázových systémů. Dílčím přínosem naplnění cíle byla definice metodiky záznamu dat ve výuce databází a výběr vhodných metod pro jejich kvantifikaci pomocí vhodných statistických metod.

Z hlediska metodiky zpracování praktické části by bylo možné praktickou rozšířit zpracování kvantitativního charakteru i o kvalitativní výzkum. V souvislosti s tím je ovšem mít na paměti vysokou komplexitu edukačních procesů, jenž probíhají a především vysoké nároky na správné stanovení vhodných metod zpracování. Kromě toho by byla realizace kvalitativního výzkumu nad údaji z celého semestru náročná i na zdroje. Výsledkem by potom ale mohlo být detailní poznání kognitivního modelu dotazování studenta. A předpokládám, že by bylo možné detailně sledovat a kategorizovat jednotlivé chyby do nových syntaktických a sémantických kategorií.

Pokud uvažujeme o dalších experimentech obdobného charakteru je nezbytné brát v patrnost možno realizace edukačního experimentu. Ten by mohl být realizován za použití odlišných výukových metod v oddělených skupinách studentů. V tom případě by ale bylo nezbytné mít dostatečný počet studentů, se kterými by experiment bylo možné provést a zároveň by bylo nezbytné provádět experiment opakovaně a zvýšit tím tak reliabilitu měření. Návrh

experimentu, jehož cílem by byla manipulace s výukovými materiály za účelem dosažení odlišného výsledku učení u studentů se ovšem potýká s mnoha problémy, které spočívají především ve spolehlivém měření výukového efektu. Částečně praktická část práce ukázala potenciál v případě, že by se ve výuce použilo různé grafické rozhraní. Již při analýze četnostní dotazů byl nalezen statisticky významný rozdíl v jejich chybovosti.

Další rozšíření výzkumu by mohlo být rozšířením testování vstupních předpokladů pomocí vstupního didaktického testu. V tom případě by analýza testu mohla být srovnána s výstupním didaktickým testem a na základě toho přesněji stanovena přesněji míra učení studenta. V korespondenci s ověřením vstupních předpokladů by tak analýza mohla poskytnou údaje o souvislosti vstupních předpokladů studenta s ostatními předpoklady při jeho vstupu do procesu učení.

Celkovou koncepci práce jsem sestavoval s minimálním omezením na určitý produkt nebo produktovou skupinu a bylo mým cílem pracovat na základě dotazovacího jazyka. Ten jsem zvolil proto, že jsem jej identifikoval jako stabilní prvků teorie databázových systémů. Potenciální uplatnění výsledků výzkumu obdobného charakteru lze nejlépe aplikovat v krátkodobém hledisku jako zpětnou vazbu do výuky nebo z dlouhodobějšího hlediska existuje možnost hledání metod pro automatické vyhodnocení práce studenta při učení nebo návrh nových metrik pro stanovení výkonu studenta. Praktická část práce může sloužit jako podklad pro hlubší bádání na poli rozvoje databázového myšlení a tvořit základ fundovanějšího experimentu prováděného s větším rozpočtem.

6.2 Závěry

Disertační práce vyslovuje tezi, že rozvoj databázového myšlení je neopomenutelnou komponentou školního vzdělávání v oblasti ICT. Na základě výsledků výzkumů lze sledovat trend umístění databází mimo ohnisko pozornosti učitelů. Jedním z důvodů může být složitost tématu, která může být bariérou při bezproblémovém seznamování s tématem. Cílem práce je tedy pokusit se najít vhodnou metodiku hodnocení výkonu rozvoje databázového myšlení, potažmo rozvoje znalosti dotazovacího jazyka u budoucích učitelů ICT.

V teoretické části disertační práce bylo v rámci hledání cílů použito standardu QCA, na jehož základě bylo možné identifikovat úrovně základního rozvoje databázového myšlení. Dále byla graficky vyjádřena vzájemná souvislost mezi praktickými činnostmi, které jsou standardem stanoveny. Z hlediska obecných kompetencí bylo databázové myšlení zařazeno do oblasti práce s informacemi. Jako hlavní cíl rozvoje databázového myšlení byl identifikován rozvoj kognitivní struktury znalostí databázového jazyka. Kromě vertikálního

členění kompetencí pro rozvoj databázového myšlení z hlediska úrovní jeho osvojení lze na rozvoj databázového myšlení pohlížet z hlediska kompetenčních skupin. Jako hlavní kompetenční skupiny byly identifikovány dovednosti spojené se zadáváním údajů do databáze, modifikace vnitřního uspořádání datové báze a tvorba výstupních reportů.

Systematizace obsahu výuky databázových systémů poskytuje přehled o hierarchii a vnitřních vazbách základů teorie databázových systémů. Kognitivní model dotazovacího jazyka je analogický s kognitivním modelem procedurálního dotazovacího jazyka. Stejně tak byla při popisu syntaktické a sémantické struktury znalosti dotazovacího jazyka použita analogie s programovacím jazykem. Výsledkem je sémantická a syntaktická struktura dotazovacího jazyka jako cíl rozvoje databázového myšlení. Databázové myšlení se zakládá na rozsáhlém teoretickém základu. Jeho strukturu lze pojmout i ve skupinách, ale za hlavní oporu při zkoumání struktury považuji ECDL. Zatímco QCA je orientována spíše na úroveň základního vzdělávání, lze ECDL kompetenčně zařadit na úroveň střední školy. Cílem kapitoly pojednávající o obsahu rozvoje databázového myšlení bylo specifikovat možnost systematizace jeho obsahu.

Pokud mluvíme o databázovém myšlení v kontextu školního vzdělávání, je nezbytné se zabývat i metodami pro jeho rozvoj. Ty se opírají o kognitivní modely neprocedurálního programování a analýzu syntaktické a sémantické struktury dotazovacího jazyka. Cílem hledání výukových metod je správná identifikace aplikačních domén kognitivního modelu rozvoje databázového myšlení studentů. V neposlední řadě existuje široká škála didaktických prostředků určených pro tuto oblast. Výsledkem rozboru metod rozvoje databázového myšlení je návrh konceptu hledání metodik pro kvantifikaci charakteristik jeho rozvoje.

Pro analýzu charakteristik rozvoje databázového myšlení studentů jsem stanovil následující základní metriky:

1) Kvantifikace rozdílu mezi použitými výukovými metodami.
2) Porovnání rozdílů ve výkonnosti studentů vyčíslením.
3) Identifikace problematických tematických celků sledováním průběhu chybovosti.
4) Určení existence vztahu mezi četností a chybovostí studentů.
5) Kvantifikace vztahu mezi četností a chybovostí studentů.
6) Detekce vstupních faktorů ovlivňujících schopnost učení studenta.

Při kvantifikaci rozdílu mezi výukovými metodami (bod 1) byl před vlastní analýzou rozdílu četnosti dotazů proveden test normality rozdělení proměnných. Nevýhodou použití statistiky

pro testování rozdílu výsledku výukových metod je možnost opomenutí vstupních proměnných, jež mohou ovlivňovat výsledky. Aplikační doménou metodiky může být výběr vhodného uživatelského rozhraní pro výuku.

Při vyčíslování rozdílu mezi výkony jednotlivých studentů (bod 2) bylo provedeno rozdělení studentů do výkonnostních skupin. To umožnilo provést oddělnou analýzu trendu chybovosti a zároveň minimalizovat vliv výskytu odlehlých hodnot při výpočtu. Pokud použijeme chybovost studentů jako hlavní ukazatel jejich výkonu, je hlavní nevýhodou signifikantní rozdíl ve výsledcích analýzy rozptylu v závislosti na použité výukové metodě. Potenciální aplikace analýzy rozdílu ve výkonnosti studentů patří v rámci konstrukce tutorského systému do oblasti zjišťování zpětné vazby. Potenciál pro rozšíření studie může být v kvalitativní analýze chyb ve výkonu studentů. Jiné praktické využití metrik učení SQL pro konstrukci automatizovaných tutorských systémů (ATS) popisuje Kenny (Kenny & Pahl, 2005). ATS může poskytovat rozšířenou zpětnou vazbu při učení dotazovacího jazyka, být tematickým průvodcem nebo generovat osobní hodnocení studenta. Na základě kvantifikace rozdílu ve výkonnosti studentů lze porovnávat aktivity jednotlivých uživatelů. Z ní pak může vyplynout konkrétní doporučení pro úpravy uživatelského rozhraní systému.

Cílem identifikace problematických tematických celků na základě sledování chybovosti (bod 3) bylo nalézt oblasti, které studentům způsobují největší potíže. Při jejich kvantifikaci byly analyzovány krajní hodnoty chybovosti. Omezení analýzy spočívá v nemožnosti rozlišení externích vlivů, které nemusí přímo souviset s obtížemi se zvládnutím tématu. Jedná se například o technické obtíže, nesrozumitelnost zadání nebo neúplné vysvětlení problému učitelem. Aplikace metodiky by přicházela v úvahu do konstrukce ATS jako modul pro určení problematického tématu určeného k procvičování. Potenciálně by identifikace náročných témat mohla zahrnovat i kvantitativní analýzy s cílem provést bližší analýzu zdrojů problémů.

Cílem výpočtu regrese kvantity procvičených dotazů a výsledné chybovosti studenta (bod 4) bylo podrobněji prozkoumat sílu jejich vztahu. V disertační práci sice vyšla velice slabá korelace, ale to lze vysvětlit vlivem odlehlých proměnných vyplývajících z technických omezení měření. Výsledek koresponduje s výchozím názorem, že čím více je procvičovaných dotazů, tím větší je možnost na zlepšení. Omezení výpočtu vyplývají přímo z výběru použité metody. Při stanovení korelace mohlo mít na výsledky vliv nerovnoměrné umístění aktivit studentů v rámci časového úseku. Tím se mohla projevit silnější korelace v momentech, kdy by mohlo být sledováno učení z chyb. Aplikace výsledků do metodiky výuky lze vyjádřit pravidlem, že větší počet praktických příkladů vede k učení, potažmo nižší chybovosti. Další rozvoj studie by mohl směřovat ke kvalitativnímu zpracování výsledků.

Pro zkoumání kvantifikace vztahu mezi četností a chybovostí studentů (bod 5) byla vybrána regresní analýza. Jejím použitím došlo ke kvantifikaci vztahu mezi četnosti a chybovosti studenta. Výsledek ukázal, že čím menší je četnost dotazů, tím větší je chybovost. To odpovídá vstupnímu předpokladu, který považuje učení dotazovacího jazyka za praktickou činnost, ve které se student zlepšuje postupným učením z chyb. Výsledky by bylo možné vyjádřit graficky a na jejich základě stanovit relaci mezi křivkou učení a kvantitou učení. Cílem analýzy vztahu mezi vstupními předpoklady a výsledky učení bylo determinovat faktory s největším vlivem na výsledky studenta. Kvantifikace byla provedena vícenásobnou analýzou rozptylu. Výsledky lze interpretovat tak, že zkušenosti se vzdělávacím softwarem a rodinou jako hlavním zdrojem poznání v oblasti databází mají vliv na výsledky učení studenta.

Hlavní cíl praktické části byl soustředěn na detekci vhodných metod pro analýzu výkonu studenta při učení dotazovacího jazyka. Cíle bylo dosaženo sběrem dat z praktické výuky a při jejich zpracování byl proveden výběr vhodných statistických metod. Cílem teoretické části bylo rozpracovat téma výuky databázových systémů z hlediska cílů, obsahu a metod výuky. Zpracováním dostupných zdrojů byl položen základ pro další bádání, který by mohl přiblížit téma širšímu povědomí českých učitelů. Jak totiž z práce vyplynulo, není téma databázových systémů na školách dostatečně akcentováno a výuka databází se prakticky nerealizuje. Je třeba soustředit pozornost na přípravu učitelů v dané oblasti a tím rozvoj databázového myšlení začlenit do školního vzdělávání.

7 Seznam tabulek

8 Seznam obrázků

9 Literatura

ASTON, M. Professional development and teacher education--Have we got it right? In *Computers & Education.* 1988, roč. 12, č. 1, s. 79-83. ISSN 0360-1315.

BANKS, J. *The history of science and technology,* Adam Matthew Publications, 1993. ISBN 1-857110528.

BELANGER, F. & CARTER, L. The impact of the digital divide on e-government use. In *Commun. ACM.* 2009, roč. 52, č. 4, s. 132-135. ISSN 0001-0782.

BELZ, H., LISÁ, D. & SIEGRIST, M. *Klíčové kompetence a jejich rozvíjení východiska, metody, cvičení a hry,* Praha: Portál, 2001. ISBN 80-7178-479-6.

BIFFAH, H., VENKY, S. & JOSE, M. Multiple representation for understanding data structures. In *Computers & Education.* 1997, roč. 29, č. 1, s. 1-11. ISSN 0360-1315.

BOTTINO, R. M. Comparing different approaches to programming from an educational viewpoint. In *Computers & Education.* 1992, roč. 18, č. 4, s. 273-281. ISSN 0360-1315.

BOULET, M.-M. Educational software design using a diagnostic approach. In *Computers & Education.* 1987, roč. 11, č. 3, s. 219-227. ISSN 0360-1315.

BOWERMAN, C. Writing and the computer: An intelligent tutoring systems solution. In *Computers & Education.* 1992, roč. 18, č. 1-3, s. 77-83. ISSN 0360-1315.

BOYCE, R. F., D. CHAMBERLIN, D., KING III, W. F. & HAMMER, M., M. Specifying queries as relational expressions: the SQUARE data sublanguage. In *Communications of the ACM.* 1975, roč. 18, č. 11, s. 621-628. ISSN 0001-0782.

BOYCE, R. F. & CHAMBERLIN, D. D. SEQUEL: A structured English query language. 1974

BOYCE, R. F., CHAMBERLIN, D. D., HAMMER, M. M. & KING, W. F. Specifying queries as relational expressions. In *SIGIR Forum.* 1974, roč. 9, č. 3, s. 31-47.

BOYD, G. M. Four ways of providing computer assisted learning and their probable impacts. In *Computers & Education.* 1982, roč. 6, č. 3, s. 305-310. ISSN 0360-1315.

BRASS, S. & GOLDBERG, C. Semantic errors in SQL queries: A quite complete list. In *Journal of Systems and Software.* 2006, roč. 79, č. 5, s. 630-644.

BRENT, M. & MITROVIC, A. *User Modeling 2005 Using Learning Curves to Mine Student Models,* 2005. 79-88 s.

BROOKS, R. E. Studying programmer behavior experimentally: the problems of proper methodology. In *Commun. ACM.* 1980, roč. 23, č. 4, s. 207-213.

BRUSILOVSKY, P., SOSNOVSKY, S., LEE, D. H., YUDELSON, M., ZADOROZHNY, V. & ZHOU, X. *An open integrated exploratorium for database courses,* Madrid, Spain: ACM, 2008. 22-26 s. ISBN 978-1-60558-078-4

BRYANT, J. H. & SEMPLE, P. *GIS and file management,* 1966. 97-107 s.

BUSH, G. Thinking Around the Corner: The Power of Information Literacy. In *Phi Delta Kappan.* 2009, roč. 90, č. 6, s. 446-447. ISSN 00317217.

CALDEIRA, C. P. *Teaching SQL: a case study,* Madrid, Spain: ACM, 2008. 340-340 s. ISBN 978-1-60558-078-4.

CALLISTER, T. A. & BURBULES, N. C. Computer literacy programs in teacher education: What teachers really need to learn. In *Computers & Education.* 1990, roč. 14, č. 1, s. 3-7. ISSN 0360-1315.

CAMSTRA, B. Make computer assisted instruction smarter. In *Computers & Education.* 1977, roč. 1, č. 3, s. 177-183. ISSN 0360-1315.

CEN, H., KOEDINGER, K. & JUNKER, B. *Intelligent Tutoring Systems Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement,* 2006.

164-175 s.
CITRON, J. Computer education in times of recession: Should Pascal come first? In *Computers & Education.* 1983, roč. 7, č. 3, s. 149-152. ISSN 0360-1315.
CODD, E. F. A relational model of data for large shared data banks. In *Commun. ACM.* 1970, roč. 13, č. 6, s. 377-387. ISSN 0001-0782.
CODD, E. F. *ACM Turing award lectures Relational database: a practical foundation for productivity,* ACM, 1981. 109 s.
COOK, W. R. High-level problems in teaching undergraduate programming languages. In *SIGPLAN Not.* 2008, roč. 43, č. 11, s. 55-58. ISSN 0362-1340.
CULBERSON, J. & RAWLINS, G. Turtlegons: generating simple polygons for sequences of angles. 1985
DEKEYSER, S., RAADT, M. D. & LEE, T. Y. *Computer assisted assessment of SQL query skills* Ballarat, Victoria, Australia Australian Computer Society, Inc., 2007 53-62 s. ISBN 1-920-68244-9.
EGAN, D. E. & GREENO, J. G. Acquiring cognitive structure by discovery and rule learning. In *Journal of Educational Psychology.* 1973, roč. 64, č. 1, s. 85-97. ISSN 1939-2176.
ENKENBERG, J. Situated programming in a legologo environment. In *Computers & Education.* 1994, roč. 22, č. 1-2, s. 119-128. ISSN 0360-1315.
FENTON, J. & BECK, K. Playground: an object-oriented simulation system with agent rules for children of all ages. In *SIGPLAN Not.* 1989, roč. 24, č. 10, s. 123-137.
FISHER, K. We need more than one: why students need a sophisticated understanding of programming languages. In *SIGPLAN Not.* 2008, roč. 43, č. 11, s. 62-65. ISSN 0362-1340.
FISCHER, G., LEMKE, A. C., MASTAGLIO, T. & MORCH, A. L. The role of critiquing in cooperative problem solving. In *ACM Trans. Inf. Syst.* 1991, roč. 9, č. 2, s. 123-151.
GAGNE, R. M. & WHITE, R. T. Memory Structures and Learning Outcomes. In *Review of Educational Research.* 1978, roč. 48, č. 2, s. 187-222.
GEORGE, B. & VALEVA, A. *A database security course on a shoestring,* Houston, Texas, USA: ACM, 2006. 7-11 s. ISBN 1-59593-259-3.
GIBSON, P. J. & O'KELLY, J. *Software engineering as a model of understanding for learning and problem solving,* 2005. ISBN 1-59593-043-4.
GÖKTEPE, M., ÖZGÜÇ, B. & BARAY, M. Design and implementation of a tool for teaching programming. In *Computers & Education.* 1989, roč. 13, č. 2, s. 167-178. ISSN 0360-1315.
GUIMARAES, M. The Kennesaw Database Courseware (KDC): strong points, weak points, and experience using it in a classroom environment In *J. Comput. Small Coll.* 2006, roč. 21, č. 3, s. 91-96. ISSN 1937-4771.
GUIMARAES, M. & MURRAY, M. *Using animation courseware in the teaching of database security,* Destin, Florida, USA: ACM, 2007. s. 253-258. ISBN 978-1-59593-920-3.
HANDLER, M. G. Preparing new teachers to use computer technology: Perceptions and suggestions for teacher educators. In *Computers & Education.* 1993, roč. 20, č. 2, s. 147-156. ISSN 0360-1315.
HAZARI, S. Computer literacy training model for higher education faculty. In *Computers & Education.* 1991, roč. 17, č. 2, s. 163-167. ISSN 0360-1315.
HENDL, J. *Přehled statistických metod zpracování datanalýza a metaanalýza dat,* Praha: Portál, 2004. ISBN 80-7178-820-1.
HOLDEN, E. P. *Assessment of an introductory database course: a case study,* Cincinnati, OH, USA: ACM, 2008. 131-138 s. ISBN 978-1-60558-329-7.
HOLLINGSWORTH, J. E. *Teaching query writing: an informed instruction approach,*

Madrid, Spain: ACM, 2008. 351 s. ISBN 978-1-60558-078-4.
HOPCROFT, J. E. & ULLMAN, J. D. *Formal languages and their relation to automata,* Addison-Wesley Longman Publishing Co., Inc., 1969. 262 s.
CHAN, H., SIAU, K. & WEI, K.-K. The effect of data model, system and task characteristics on user query performance: an empirical study. In *SIGMIS Database.* 1997, roč. 29, č. 1, s. 31-49.
CHAN, H. C., WEI, K. K. & SIAU, K. L. User-Database Interface: The Effect of Abstraction Levels on Query Performance. In *MIS Quarterly.* 1993, roč. 17, č. 4, s. 441-464.
CHRISMAN, C. *Teaching Database design through an Entity-Relationship approach* Indianapolis, Indiana, United States ACM, 1982 4-7 s. ISBN 0-89791-067-2.
JARKE, M. & VASSILIOU, Y. A framework for choosing a database query language. In *ACM Comput. Surv.* 1985, roč. 17, č. 3, s. 313-340.
JUKIC, N. & GRAY, P. Using real data to invigorate student learning. In *SIGCSE Bull.* 2008, roč. 40, č. 2, s. 6-10. ISSN 0097-8418.
KALHOUS, Z. & OBST, O. *Školní didaktika,* Praha: Portál, 2002. ISBN 80-7178-253-X.
KEARNS, R., SHEAD, S. & FEKETE, A. *A teaching system for SQL,* 1996. ISBN 0-89791-958-0.
KENNY, C. & PAHL, C. *Automated tutoring for a database skills training environment,* St. Louis, Missouri, USA: ACM, 2005. 58-62 s. ISBN 1-58113-997-7.
KHOSROW-POUR, M. *Encyclopedia of Information Science and Technology,* London: Idea Group Reference, 2005. 3121 s. ISBN 1-59140-553-X.
KONG, S. C. & LI, K. M. Collaboration between school and parents to foster information literacy: Learning in the information society. In *Computers & Education.* 2009, roč. 52, č. 2, s. 275-282. ISSN 0360-1315.
KURTZ, B. L. & ADAMS, J. M. Using concept expansion and level integration in an introductory computer science course. In *SIGCSE Bull.* 1988, roč. 20, č. 1, s. 244-248.
KUSHAN, B. *Preparing programming teachers,* ACM New York: New York, 1994. ISBN 0-89791-646-8.
LAYMAN, J. & HALL, W. Logo: A cause for concern. In *Computers & Education.* 1988, roč. 12, č. 1, s. 107-112. ISSN 0360-1315.
LEAVENOWORTH, B. M. & SAMMET, J. E. An overview of nonprocedural languages. In *ACM SIGPLAN Notices.* 1974, roč. 9, č. 4, s. 1-12. ISSN 0362-1340.
LEVIEN, R. E. & MARON, M. E. *A computer system for inference execution and data retrieval,* 1967. s. 715-721. ISBN 064612448X.
MA, J. Problem-based learning with database systems. In *Computers & Education.* 1994, roč. 22, č. 3, s. 257-263. ISSN 0360-1315.
MAŇÁK, J. & ŠVEC, V. *Výukové metody,* Brno: Paido, 2003. ISBN 80-7315-039-5 (váz.) AN - L_NK_SKC.
MAYER, E. Putting general education to work: The key competencies report. In *Melbourne, Vic: Australian Education Council and Ministers for Vocational Education Employment and Training.* 1992
MAYER, R. E. Acquisition processes and resilience under varying testing conditions for structurally different problem-solving procedures. In *Journal of Educational Psychology.* 1974, roč. 66, č. 5, s. 644-656.
MAYER, R. E. Information Processing Variables in Learning to Solve Problems. In *Review of Educational Research.* 1975, roč. 45, č. 4, s. 525-541.
MAYER, R. E. A psychology of learning BASIC. In *Commun. ACM.* 1979, roč. 22, č. 11, s. 589-593. ISSN 0001-0782.
MAYER, R. E. The Psychology of How Novices Learn Computer Programming. In *ACM Computing Surveys.* 1981, roč. 13, č. 1, s. 121-141.

MAYER, R. E. & BAYMAN, P. Using conceptual models to teach BASIC computer programming. In *Journal of Educational Psychology.* 1988, roč. 80, č. 3, s. 291-298.
MAYER, R. E. & GREENO, J. G. Structural differences between outcomes produced by different instructional methods. In *Journal of Educational Psychology.* 1972, roč. 63, č. 2, s. 165-173.
MCINTYRE, D. R., PU, H.-C. & WOLFF, F. G. Use of software tools in teaching relational database design. In *Computers & Education.* 1995, roč. 24, č. 4, s. 279-286. ISSN 0360-1315.
MEEKER, R. & NOHL, D. Using a practicum experience in your database course In *J. Comput. Small Coll.* 2007, roč. 23, č. 1, s. 91-96. ISSN 1937-4771.
MILNER, S. D. & WILDBERGER, A. M. Determining appropriate uses of computers in education. In *Computers & Education.* 1977, roč. 1, č. 2, s. 117-123. ISSN 0360-1315.
MITROVIC, A. Learning SQL with a computerized tutor. In *SIGCSE Bull.* 1998, roč. 30, č. 1, s. 307-311.
MUDRÁK, D. *Rozvíjení kompetence pro manipulaci se strukturami jako součást informační výchovy,* Praha: PedF UK, 2007. 166 s.
MURRAY, M. & GUIMARAES, M. Expanding the database curriculum. In *J. Comput. Small Coll.* 2008, roč. 23, č. 3, s. 69-75. ISSN 1937-4771.
NCVER. *Defining generic skills: at a glance.* 2003. [Citováno: 4. 9. 2009]. Dostupné z: <http://www.ncver.edu.au/research/proj/nr2102b.pdf>.
NEUMAJER, O. *ICT kompetence učitelů,* Praha: PedF UK, 2007. 167 s.
OECD. *The definition and selection of key competencies*. 2005. [Citováno: 6. 9. 2009]. Dostupné z: <http://www.oecd.org/dataoecd/47/61/35070367.pdf>.
OGDEN, W. C. Implications of a cognitive model of database query: Comparsion of a natural language, formal language and direct manipulation interface. In *SIGCHI Bull.* 1986, roč. 18, č. 2, s. 51-54.
OGDEN, W. C. & BROOKS, S. R. Query languages for the casual user: Exploring the middle ground between formal and natural languages. 1983
PAPERT, S. *Mindstorms,* New York: Basic Books, Inc., Publishers, 1980. 230 s. 1st. edition. ISBN 0-465-04627-4.
PEELLE, H. A. Computer metaphors: Approaches to computer literacy for educators. In *Computers & Education.* 1983, roč. 7, č. 2, s. 91-99. ISSN 0360-1315.
PRIOR, C. J. *Online assessment of SQL query formulation skills,* 2003. ISBN 0-909925-98-4.
PRŮCHA, J., WALTEROVÁ, E. & MAREŠ, J. *Pedagogický slovník,* Praha: Portál, 2003. ISBN 80-7178-772-8.
PYRCZAK, F. Development of diagnostic tests for computer literacy. In *Computers & Education.* 1990, roč. 14, č. 3, s. 213-216. ISSN 0360-1315.
QCA, Q. A. C. A. *Information and Communication Technology Teacher's guide,* London: Qualifications and Curriculum Authority, 2003. 73 s. ISBN 1-85838-333-1.
QCA, Q. A. C. A. *Framework for teaching ICT capability: Years 7, 8 and 9,* London: Qualifications and Curriculum Authority, 2007. 71 s.
RADA, E. P. A. *Doporučení Evropského parlamentu a Rady ze dne 18. prosince 2006 o klíčových schopnostech pro celoživotní učení*. 2006. [Citováno: 4. 9. 2009]. Dostupné z: <http://eur-lex.europa.eu/LexUriServ/.../l_39420061230cs00100018.pdf>.
RAE, J. Getting to grips with database design: A step by step approach. In *Computers & Education.* 1990, roč. 14, č. 6, s. 481-488. ISSN 0360-1315.
RAFOTH, M. A. *Encyclopedia of Applied Psychology,* New York: Elsevier, 2004. 743 s. ISBN 978-0-12-657410-4.
RAMBOUSEK, W. & ŠTÍPEK, J. *Výzkum informační výchovy na základních školách,* Plzeň: Koniáš, 2007. 360 s. 1. ISBN 80-86948-10-2.

REISNER, P. Human Factors Studies of Database Query Languages: A Survey and Assessment. In *ACM Comput. Surv.* 1981, roč. 13, č. 1, s. 13-31.
RESNICK, M. Behavior construction kits. In *Commun. ACM.* 1993, roč. 36, č. 7, s. 64-71.
RÝDL, K., KOŠŤÁLOVÁ, H., PÍŠOVÁ, M., SPILKOVÁ, V., STÝBLOVÁ, H. & TOMÁŠEK, F. *Tvorba profesního standardu kvality učitele*. 2009. [Citováno: 15.5.2009]. Dostupné z: <http://www.msmt.cz/uploads/soubory/zakladni/VS_Tvorbastandardu_vstupnidokumentproverejnoudiskusi.pdf>.
SAJ-NICOLE, J., ELLIOT, S., ROBERT, G. & KATE, E. Just so stories: how the program got that bug. In *SIGCUE Outlook.* 1983, roč. 17, č. 4, s. 13-26.
SAUTER, V. L. Predicting computer programming skill. In *Computers & Education.* 1986, roč. 10, č. 2, s. 299-302. ISSN 0360-1315.
SELF, J. A. Student models and artificial intelligence. In *Computers & Education.* 1979, roč. 3, č. 4, s. 309-312. ISSN 0360-1315.
SEYED-ABBASSI, B. A SQL project as a learning method in a database course. 1993
SHIVERS, O. Why teach programming languages. In *SIGPLAN Not.* 2008, roč. 43, č. 11, s. 130-132. ISSN 0362-1340.
SHNEIDERMAN, B. Teaching programming: A spiral approach to syntax and semantics. In *Computers & Education.* 1977, roč. 1, č. 4, s. 193-197. ISSN 0360-1315.
SHNEIDERMAN, B., MAYER, R., MCKAY, D. & HELLER, P. Experimental investigations of the utility of detailed flowcharts in programming. In *Commun. ACM.* 1977, roč. 20, č. 6, s. 373-381.
SCHIBECI, R. A. Logo in pre-service and in-service teacher education. In *Computers & Education.* 1990, roč. 14, č. 1, s. 53-60. ISSN 0360-1315.
SCHLAGER, M. S. & OGDEN, W. C. A cognitive model of database querying: a tool for novice instruction. In *SIGCHI Bull.* 1986, roč. 17, č. 4, s. 107-113.
SIMMONS, R. F. Answering English questions by computer: a survey. In *Commun. ACM.* 1965, roč. 8, č. 1, s. 53-70.
SKALKOVÁ, J. *Obecná didaktika,* Praha: ISV, 1999. ISBN 80-85866-33-1 (váz.) AN - L_NK_SKC.
SLAGLE, J. R. Experiments with a deductive question-answering program. In *Commun. ACM.* 1965, roč. 8, č. 12, s. 792-798.
SMITH, D. J. & SAGE, M. W. Computer literacy and the education/training interface. In *Computers & Education.* 1983, roč. 7, č. 4, s. 227-234. ISSN 0360-1315.
SOLOWAY, E. Learning to program = learning to construct mechanisms and explanations. In *Commun. ACM.* 1986, roč. 29, č. 9, s. 850-858.
SPARER, E. *Sinclair Logo 1 Turtle Graphics,* Cambridge: Sinclair Reserach Ltd, 1984. 94 s. ISBN 185016 018 X.
SURAWEERA, P. & MITROVIC, A. *Intelligent Tutoring Systems KERMIT: A Constraint-Based Tutor for Database Modeling,* 2002. 377-387 s.
ŠTÍPEK, J. *Informační elementaristika a systémové pojetí jejího obsahu,* Praha: PedF UK, 2004.
THORNBURG, D. D. From metaphors to microworlds. The challenge of creating educational software. In *Computers & Education.* 1988, roč. 12, č. 1, s. 11-15. ISSN 0360-1315.
TOMEK, I. *Josef, programming for everybody,* 1982. 188-192 s. ISBN 0-89791-067-2.
TURNER, J. A., JARKE, M., A. STOHR, E., VASSILIOU, Y. & H. WHITE, N. *Coupling field studies with laboratory experiments for the evaluation of computer languages,* 1985. ISBN 0-89791-150-4.
UDOH, E. *Teaching database in an integrated oracle environment,* Bologna, Italy: ACM, 2006. 71-74 s. ISBN 1-59593-603-3.

UENO, M., NISHIKI, T. & TSUSHIMA, K. *NeGAS: logo based authoring environment for 3D computer animation,* 2006. ISBN 1-59593-380-8.

UNDERWOOD, J., SPAVOLD, J. & UNDERWOOD, G. Novice use of a relational database: A case study of a local history data file. In *Computers & Education.* 1990, roč. 15, č. 1-3, s. 227-232. ISSN 0360-1315.

VÚP. *Rámcový vzdělávací program pro předškolní vzdělávání*. 2004. [Citováno: 4. 9. 2009]. Dostupné z: <http://www.rvp.cz/soubor/RVP_PV-2004.pdf>.

VÚP. *Klíčové kompetence v základním vzdělávání*. 2007a. [Citováno: 4. 9. 2009]. Dostupné z: <http://www.vuppraha.cz/soubory/kkzv.pdf>.

VÚP. *Rámcový vzdělávací program pro gymnázia,* Praha: Výzkumný ústav pedagogický v Praze, 2007b. 100 s. ISBN 978-80-87000-11-3.

VÚP. *Rámcový vzdělávací program pro základní vzdělávání*. 2007c. [Citováno: 4. 9. 2009]. Dostupné z: <http://www.vuppraha.cz/soubory/RVPZV_2007-07.pdf>.

WEERASINGHE, A. & MITROVIC, A. *Knowledge-Based Intelligent Information and Engineering Systems Using Affective Leaner States to Enhance Learning,* 2005, s. 465-471.

WEERASINGHE, A., MITROVIC, A. & MARTIN, B. A Preliminary Study of a General Model for Supporting Tutorial Dialogues. *ICCE 2008.* Taiwan, 2008.

WEINBERG, G. M. *The Psychology of Computer Programming,* John Wiley \& Sons, Inc., 1985. 304 s. ISBN 0442292643.

WELTY, C. & STEMPLE, D. W. Human factors comparison of a procedural and a nonprocedural query language. In *ACM Trans. Database Syst.* 1981, roč. 6, č. 4, s. 626-649. ISSN 0362-5915.

WHITTAKER, S. & STENTON, P. User studies and the design of natural language systems. In *EACL '89 Proceedings.* 1989, roč. 116-123.

WIKIPEDIA. *Timeline of programming languages*. 2008. [Citováno: 20. 7. 2008]. Dostupné z: <http://en.wikipedia.org/wiki/Timeline_of_programming_languages>.

WILSON, D. J. *Entity-relationship diagrams and English: an analysis of some problems encountered in a database design course,* 1987, s. 26-35. ISBN 0-89791-217-9.

WILSON, R. A. & KEIL, F. C. *The MIT Encyclopedia of the Cognitive Sciences,* MIT Press, 2001. ISBN 0-262-73144-4.

WU, L. *Error recovery in human-computer interaction: a preliminary study in a database learning environment,* Napa Valley, California, USA: ACM, 2008, s. 93-96. ISBN 978-1-60558-257-3.

YAMAMOTO, N., NESBIT, J. C. & NAKAYAMA, K. Procedure definition and higher-order programming in Hyperlogo. In *Computers & Education.* 1991, roč. 17, č. 2, s. 155-162. ISSN 0360-1315.

YEHOSHAFAT, S. G. O. Teaching recursive programming using parallel multi-turtle graphics. In *Computers & Education.* 1991, roč. 16, č. 3, s. 267-280. ISSN 0360-1315.

10 Přílohy

10.1 Vstupní dotazník

Otázka	Text otázky	Odpověď
1	Počítač mám pro sebe dostupný	Doma
		Ve škole
		Jinde
2	Počítač používám již	Méně než jeden rok
		Jeden až tři roky
		Tři až pět let
		Pět až deset let
		Více než deset let
3	Počítač používám především pro	Vyhledávání lidí, věcí a myšlenek na internetu
		Hraní počítačových her
		Psaní textových dokumentů
		Spolupráci s lidmi a týmy po internetu
		Práci s tabulkovým procesorem
		Stahování software (i her)
		Kreslení, malování nebo používání grafických programů
		Práci se vzdělávacím softwarem
		Internet pro stahování muziky
		Internet pro elektronickou komunikaci (např.: e-mail, chat)
4	Dobře se vyznám a umím	Spustit počítačovou hru
		Zbavit počítač virů
		Otevřít soubor
		Vytvořit / upravit dokument
		Použít databázový program k vytvoření seznamu adres
		Zkopírovat soubor z diskety
		Vytvořit počítačový program
		Použít tabulkový kalkulátor k vytvoření grafu
		Vytvoření prezentace
		Vytvořit WWW stránku
5	O mých zkušenostech s počítačem lze říci	Je pro mě velice důležité pracovat s počítačem
		Myslím si, že hraní a práce s počítačem je zábava
		Používám počítač, protože mě to velice zajímá
		Ztrácím pojem o čase, když pracuji s počítačem
6	Kdo Vás naučil nejvíce z toho, co víte o počítačích a jejich používání?	Základní škola
		Střední škola
		Vysoká škola
		Moji přátelé
		Moje rodina
		Naučil jsem se sám
		Jinde
7	Kdo Vás naučil nejvíce z toho, co víte o internetu a jeho používání?	Základní škola
		Střední škola
		Vysoká škola

Otázka	Text otázky	Odpověď
		Moji přátelé
		Moje rodina
		Naučil jsem se sám
		Jinde
8	S databázovými systémy	jsem se ještě nesetkal(a)
		mám základní znalost práce s databázemi
		jsem pokročilý uživatel databázových systémů
		jsem databázový expert
9	Již před tímto kurzem jsem znal(a) tyto databázové produkty	MySQL
		Postgresql
		Oracle
		Microsoft SQL Server
		Informix
		SyBase
		InterBase
10	V předmětu Databázové a informační systémy bych se chtěl	Naučit se pracovat s databází
		Naučit se projektovat informační systém
		Naučit se pracovat s CASE nástrojem
		Naučit se SQL

Tabulka 34 Vstupní dotazník

10.2 Databázové kompetence dle QCA

Název aktivity: 1A Rozšiřování slovní zásoby
Výukový cíl: Žáci se naučí sdělovat své myšlenky vybíráním a přidáváním vhodných slov z textu.
Výuková činnost: Učitelé ukážou třídě obrázky s dvojrozměrnými a trojrozměrnými geometrickými tvary. Poté nechají žáky prohlédnout slovní banku obsahující slova, která by mohli potřebovat k popisu těchto tvarů. Následně budou učitelé prezentovat tvary celé třídě. Při prezentaci tvarů popíšou vlastnosti jednotlivých tvarů a nechají žáky zopakovat popis tvarů vlastními slovy. Upozorní je na klíčová slova často se vyskytující ve slovních bankách a nechají je přidat vlastní slova do své slovní zásoby. Popis tak může například vypadat: „Toto je obdélník, který má čtyři strany.“ Dvojrozměrný nebo trojrozměrný tvar může být přidán do slovní banky, a to tak, aby žáci mohli vybrat nejenom text, ale i obrázek. Žáci svou práci mohou s pomocí učitele vytisknout.
Výsledek výuky: Cílem výše uvedené činnosti je nechat žáky vytvořit jasný text bez chyb, jenž popisuje vlastnosti dvojrozměrného nebo trojrozměrného tvaru.
Poznámka: Ve výuce lze použít software, například Clicker Grids, což je multimediální nástroj pro podporu psaní pro žáky. Při práci v něm je obrazovka rozdělena dvě části. V horní části se nachází textový procesor Clicker Writer a ve spodní části obrazovky je Clicker Grid, ve kterém jsou buňky obsahující písmena, slova nebo fráze, na které lze kliknout, a tím je poslat do Clicker Writer. To umožňuje žákům psát věty, aniž by museli znát abecedu a dovedli psát na klávesnici. Slova v Clicker Gridu jsou získávána ze slovní banky, se kterou může žák libovolně manipulovat. Dovoluje to tak žákům mít vlastní slovníček obsahující vlastní obrázky. Zařízení pro import obrázků do programu může dovolit učiteli zajistit, že se žáci budou pohybovat v rámci vybraného spektra. Například tak může omezit spektrum na rovnoramenné, nerovnostranné, tupé a pravoúhlé trojúhelníky. U nich mohou žáci přidat svůj popis. Základní výuka může být základem pro podporu činností v jednotce 1D Vlastnosti a kategorie, ve které mohou žáci použít stejný princip k označení nebo třídění věcí.

Tabulka 35 Činnost 1A Rozšiřování slovní zásoby

Název aktivity: 1B Informace kolem nás
Výukový cíl: Cílem je předvést, že informace mohou být prezentovány v různých podobách a mohou být získávány z různých zdrojů.
Výuková činnost: Učitel použije sbírku hudebních nástrojů a nahrávek zvuků hudebních nástrojů, aby žákům přehrál zvuk každého hudebního nástroje. Potom je nechá spojit zvuk hudebního nástroje s obrázkem hudebního nástroje. Promluví s žáky o tom, že některé hudební nástroje, například klavír či viola, mají řadu zvuků, zatímco zvuk jiných, např.: buben, triangl, se liší v síle zvuku od hlasitého po tichý. Povzbudí žáky v přemýšlení o tom, jaké informace můžeme získat z hraní na hudební nástroj místo koukání na jeho obrázek, například: „Zvuk může nástroj vydávat pouze tím, že na něj zahrajeme.“ (nebo poslechem jeho nahrávky). Nechá žáky vytvořit jednoduchou prezentaci o zvucích a hudebních nástrojích, za pomoci různých informačních zdrojů.
Výsledek výuky: Přínosné je pro žáky zjištění, že mohou prezentovat informace v různých podobách a sbírat informace z celé řady zdrojů.
Poznámka: Žáci mohou sami sebe nahrát do počítače nebo na kazetu a z nahrávek se pokusit poznávat spolužáky. Existují softwarové balíčky, které umožňují žákům vkládat k nástrojům poznámky nebo poslouchat zvuky, které vydávají. Existuje spoustu webových stránek, které žákům dovolí poslouchat zvuky a nechají je hádat, jaký nástroj hraje.

Tabulka 36 Činnost 1B Informace kolem nás

Název aktivity: 1C Popis a kategorizace předmětu
Výukový cíl: Žáci by se měli naučit popsat předměty pomocí klíčových slov.
Výuková činnost: Učitel připraví slovní zásobu na téma předměty ve třídě. S žáky prodiskutuje, k jakému účelu jednotlivé předměty slouží. Nechá žáky věci rozdělit do skupin podle různých hledisek. Nechá je poté odůvodnit, proč použili daný název skupiny. Nechá žáky napsat jednoduché názvy skupin nebo návodů k použití výběrem slov ze slovní banky.
Výsledek výuky: Žáci zjistí, že věci mají vlastnosti. Jsou schopni o věcech podat informace napsáním jednoduchých instrukcí a skupin pro použití ve třídě.
Poznámka: Někteří žáci budou schopni seskupit objekty podle jejich použití a poté skupině dají název v podobě klíčového slova. Hlavní nápady a techniky z této učební jednotky mohou být základem pro rozvoj znalostí z jednotek 2B „Otázky a odpovědi“ a v jednotce 3A „Úvod do databází“.

Tabulka 37 Činnost 1C Popis předmětu

Název aktivity: 1D Vlastnosti a kategorizace předmětu

Výukový cíl: Popsat věci pomocí klíčových slov.
Výuková činnost: Nechat žáky posbírat skupinu věcí ve třídě. Prodiskutovat, z jakého materiálu jsou. Zeptat se na další věci vyrobené z daného materiálu. Nechat žáky použít různá přídavná slova k popisu materiálů a objektů. Mohou použít slovní zásobu nebo přídavná jména materiálu. Nechat žáky seřadit slova reprezentující věci do tabulky nebo seznamu podle jejich materiálu.
Výsledek výuky: Žáci zjistí, že věci mají vlastnosti. Budou umět věci seskupovat do skupin podle jejich materiálu.
Poznámka: Hlavní nápady a techniky z této učební jednotky mohou být základem pro rozvoj kompetencí z jednotky 2B „Otázky a odpovědi“ a z jednotky 3A „Úvod do databází“.

Tabulka 38 Činnost 1D Vlastnosti a kategorizace předmětu

Název aktivity: 2A Vyhledání informace
Výukový cíl: Použití vhodných technik vyhledávání za účelem nalezení informace. Směřovat požadavky přímo.
Výuková činnost: Sdělit žákům, že budou používat elektronický slovník, aby hledali definice. Tím si procvičí svoje čtenářské dovednosti. Učitel připraví seznam obsahující podstatná jména, přídavná jména a slovesa za použití interaktivní tabule nebo datového projektoru. Učitel předvede, jak hledat podle abecedy, používat tlačítka zpět, domů a jak se vrátit na obsah. Učitel vysvětlí, že některé pojmy ve slovníku mohou odkazovat na jiné pojmy. Předvede, jak různě může být informace spojena s jinou. Vede žáky k tomu, aby poznali, která slova mohou být nalezena abecedním vyhledáváním a která podle slovního spojení. Vede žáky k seskupování vyhledaných definic podle různých kritérií. Někteří žáci je budou schopni seřadit podle prvního písmena anebo podle prvních dvou, tří písmen. Když budou žáci hotovi, nechá je nahlas přečíst své definice. Úkol pro celou třídu může být porovnávání vyhledávacích technik a vyhodnocení nejefektivnější techniky pro vyhledání informací.
Výsledek výuky: Žáci budou umět použít odpovídající techniku a pomocí nejpřímější techniky naleznou informace v elektronickém slovníku.
Poznámka: Někteří žáci budou schopni najít slovní spojení. Elektronický slovník s uloženými namluvenými pojmy může pomoci žákům sledovat text a porozumět známým slovům.

Tabulka 39 Činnost 2A Vyhledání informace

Název aktivity: 2B Otázky a odpovědi
Výukový cíl: Žáci se naučí připravit data pro databázi. Naučí se vyhledat odpověď na konkrétní otázku v jednoduché databázi. Žák bude schopen předložit výsledky svých zjištění.
Výuková činnost: Učitel shromáždí soubor geometrických tvarů různých velikostí a barev. Geometrické tvary ukáže žákům a nechá je vybrat geometrický tvar, u kterého budou schopni určit alespoň dvě vlastnosti (má šest stran, je kulatý, je červený atp.). Žák určený učitelem schová vybraný předmět do tašky. Ostatní žáci pokládají otázky, aby získali dostatek informací pro uhádnutí geometrického tvaru. Žáci mohou pokládat otázky, na které je možné odpovědět ano nebo ne. Mohou ale pokládat i otázky, na které lze odpovědět například počtem barev. Učitel připraví datový soubor s omezeným počtem polí, do kterého pomůže žákům zadat několik málo údajů o geometrických tvarech. Dále učitel připraví seznam otázek, které budou žáci používat při hledání odpovědi na konkrétní otázky, např.: „Jakou barvu má trojúhelník? Kolik stran má?“. Zahrne i otázku, pro kterou nejsou dostupné údaje. Žáci pracují ve dvojici a pokouší se získat odpovídající záznam.
Výsledek výuky: Pro žáky bude přínosem schopnost schraňovat informace pro databázi. Žáci budou umět používat jednoduchý vyhledávací nástroj pro hledání odpovědí na jednoduchou otázku. Žáci porozumí důvodu, proč některé otázky nemohou být zodpovězeny, pokud v databázi nejsou uloženy relevantní údaje.
Poznámka: Před výukou učitel zopakuje žákům potřebné matematické termíny. Úloha může být upravena tak, aby žáci z nižšího ročníku mohli tvary rovnou přiřadit položením otázek. Například „Má tvar více než 4 strany?“ nebo „Vyslovíš počet barev, pokud budeš počítat od 10 do 0?“. Učitel použije pro podporu výuky jednoduchý databázový software.

Tabulka 40 Činnost 2B Otázky a odpovědi

Název aktivity: 3A Úvod do databází
Výukový cíl: Žák ovládá uspořádání a třídění informace v databázi a dovede svá zjištění předvést.
Výuková činnost: Učitel jako součást vědního tématu vysvětlí třídě látku na téma materiály a ukáže, jak vytvořit databázi obsahující informace o předmětech v kuchyni. Nechá je vymyslet otázky, na které by se chtěli zeptat. Například: „Jaký je tvůj nejoblíbenější materiál? Které věci jsou pevné? Které objekty jsou pevné a silné?“. Učitel poskytne řadu různých věcí tak, aby se jich žáci mohli doma dotknout a prozkoumat je. Pomůže jim k tomu, aby mohli mluvit o vlastnostech materiálů, např. o tvrdosti, pevnosti, ohebnosti, měkkosti anebo lesklosti. Otázky jsou uzavřeného typu s odpovědí typu ano/ne. Žáci sami doplní ke každému předmětu další údaje. Jakmile žáci zahrnou všechny předměty, sestaví otázky pro celou třídu. Diskutujte s žáky o tom, jak mohou být přeloženy do vyhledávacích kritérií. Žáci se naučí z datové báze vytvořit sestavu obsahující grafy.
Výsledek výuky: Žáci budou schopni posbírat relevantní informace, zadat je do databáze a použít databázi k zodpovězení jednoduchých otázek.

Název aktivity: 3A Úvod do databází
Poznámka: Žáci, pro které je výuka obtížná, mohou dostat jednoduchou otázku. Například „Jaký je nejběžnější materiál?“. Schopnější žáci mohou dostat otázky, které budou zahrnovat vytvoření sloupcových grafů. Například tak žák může podpořit své tvrzení, že všechny kovové objekty ve sbírce jsou hladké nebo lesklé. Nebo dokázat obecnější hypotézu, že kovové materiály jsou vždy pevné. Žáci mohou použít digitální fotoaparát, aby vyfotografovali věci, které vloží do datové báze, pokud umožňuje zaznamenat grafiku. Tato aktivita staví na klíčových kompetencích zahrnutých v jednotce 1A „Rozšiřování slovní zásoby“ a 1D „Vlastnosti a kategorie“. Jednotka 2B „Otázky a odpovědi“.

Tabulka 41 Činnost 3A Úvod do databází

Název aktivity: 4A Větvené databáze
Výukový cíl: Žák se naučí organizovat, reorganizovat a analyzovat informace.
Výuková činnost: Učitel ukáže žákům způsob, jak vytvořit větvenou databázi. Pro lepší představu učitel ukáže žákům plakát, na kterém bude zobrazeno hierarchické členění živočišných druhů. Učitel použije větvenou databázi k identifikaci manželek českého krále Karla IV. Učitel připomene žákům, že na každém uzlu větvené databáze budou potřebovat odpověď na otázku typu ano nebo ne. Učitel diskutuje s žáky o relevantních typech otázek. Například „Byla připravena o hlavu?“. Žáci pracují ve dvojicích a vytvoří svoji větvenou databázi žen krále Karla IV. Žáci si navzájem otestují databáze.
Výsledek výuky: Žáci budou schopni vytvořit větvenou databázi, která identifikuje manželky Karla IV.
Poznámka: Jednotka rozvíjí klíčové myšlenky a techniky v návaznosti na jednotky 1B „Používání slovní zásoby“, 1D „Označování a třídění“, „2E Otázky a odpovědi“ a „3C Úvod do databází“. Některé programy pro větvené databáze dovolují importovat obrázky. Žáci mohou použít obrázky, které jim doporučí učitel. Může být výhodou žákům obrázky předem připravit.

Tabulka 42 Činnost 4A Větvené databáze

Název aktivity: 5A Analýza dat
Výukový cíl: Žák umí pro nalezení informace formulovat složitou otázku. Cílem výuky je naučit žáky používat složité dotazy pro nalezení informace.
Výuková činnost: Cílem je naučit žáky používat složité dotazy k nalezení informace. Dílčím cílem je naučit žáky používat ICT pro ověření hypotézy. V rámci procvičování znalosti počtů nechá učitel třídu nakreslit na papír mřížku a do ní uspořádat dvojrozměrné tvary. Pro každý tvar žák vymyslí fiktivní jméno. Žák si prohlédne a zaznamená vlastnosti každého tvaru. Žáci společně navrhnou databázi tak, aby jí bylo možné klást otázky o tvarech. Například: „Jak se jmenuje tento tvar? Kolik má vnitřních pravých úhlů? Vypadá stejně, i když se otočí o 90 stupňů? Kolik má ostrých vnitřních úhlů? Kolik má tupých úhlů? Kolik má symetrických středů? Kolik má hran? Je mozaikový?“. Pole v databázi mohou obsahovat údaje o smyšleném názvu, pravých úhlech, ostrých úhlech, tupých úhlech, čárech zrcadlení, hranách anebo mozaikách. Žáci dostanou úkol, aby vkládali podrobnosti o jednotlivých tvarech do databáze podle svých kreseb. Učitel vysvětlí třídě, které otázky mohou být zodpovězeny hledáním v souboru. Například: „ Kolik tvarů má pravý úhel? Kolik tvarů s pravým úhlem je kostkovaných?“
Výsledek výuky: Žáci budou umět zúžit složité vyhledávání. Zjistí, že položením otázky o všech záznamech v souboru mohou ověřit svůj předpoklad týkající se podmnožiny tvarů.
Poznámka: Pro ty žáky, pro které je zadání příliš náročné, může mít být úkol zjednodušen omezením pouze na trojúhelníky. Tím budou mít žáci možnost pracovat pouze se čtyřmi jedinečnými typy trojúhelníky: rovnostranný, nerovnostranný, pravoúhlý a rovnoramenný. Žáci je mohou sestrojit a dát každému smyšlené jméno. Žák přidá do databáze čtyři záznamy a oznámí smyšlené jméno jednoho typu trojúhelníku spolužákovi. Spolužák musí v databázi vyhledat smyšlený typ trojúhelníku. Náročnost úlohy lze naopak zvýšit omezením počtu hledání.

Tabulka 43 Činnost 5A Analýza dat

Název aktivity: 6A Použití internetu pro hledání ve velkých databázích a k interpretaci informací.
Výukový cíl: Žáci by se měli naučit používat složité vyhledávací dotazy k nalezení informací. Žáci se naučí spolupracovat s ostatními na interpretaci informací. Žáci budou umět hledat informace definicí různých pohledů a naučí se ověřovat věrohodnost informačního zdroje. Žáci se naučí používat ICT k organizaci a prezentaci informací tak, aby byly přizpůsobeny pro konkrétní posluchače.

Výuková činnost: Činnost může být součástí tématu Země, Slunce a Měsíc. Žáci mohou použít internet k hledání různých informací. Například: „Co je helium? Kde a jak se vytváří? Jak vypadá Země, když se na ni díváme z Měsíce? Co by sis pamatoval z procházky po Měsíci? Myslíš si, že je Slunce magnetické se severním a jižním pólem? Proč?“. Učitel zadá žákům vyhledat informace a odpovědi na tyto otázky na dvou nebo třech vybraných stránkách. Nechá je připravit prezentaci svého zjištění. Kromě toho musejí žáci být schopni porovnat kvalitu informací poskytovaných stánek. Připomeňte žákům možnost hledat v rámci vybraných stránek klíčová slova. Učitel nechá žáky pracovat ve skupině, aby nalezli a interpretovali informace. Žáci potřebují podpořit své zjištění odkazem na jiné zdroje. Učitel projde s celou třídou stránky, které používali. Dále kladem žákům otázky: „Proč byla vybrána právě tato prezentace? Jak dlouho trvalo nalezení vhodné informace? Jak se lišily údaje na jednotlivých webových prezentacích a proč? Byla nalezená informace relevantní a přijatelná?“
Výsledek výuky: Žáci se naučí nalézt, porozumět a interpretovat informace. Naučí se používat různé zdroje k ověření platnosti informací a rozpoznat různé pohledy a dopad nesprávných údajů. Žáci se naučí prezentovat své nápady a informace ve stylu, který odpovídá jejich posluchačům.
Poznámka: Tato aktivita používá klíčové znalosti a dovednosti z jednotky „6A multimediální prezentace“. Na internetu existuje spoustu informací ohledně slunečné soustavy a Země, Slunce a Měsíce. Náhodné prohledávání internetu může být v tomto případě neefektivní a může vyústit v příliš mnoho irelevantních odkazů a vysoce technických informací. Aby se tomu předešlo, může učitel připravit a vybrat stránky, které smí žáci použít před výukou.

Tabulka 44 Činnost 6A Použití internetu k prohledávání rozsáhlých databází

10.3 Databázové kompetence dle ECDL

Cílem databázového modulu ECDL je zjistit, zda uchazeč ovládá následující kompetence:
chápe, co je to databáze, jaká je její struktura a jak se s ní pracuje,
vytvoří jednoduchou databázi a umí prohlížet její vnitřní uspořádání,
vytvoří tabulku, definuje a upravuje pole a vlastnosti polí tabulky
dovede do tabulky zadávat data a následně je měnit,
umí v souladu se zadáním vytvořit, upravovat a spouštět databázové dotazy,
dovede řadit a filtrovat data v tabulce a dovede používat formuláře,
zná principy, na kterých funguje formulář,
umí vytvořit formulář pro zadávání, úpravy a odstraňování údajů v tabulkách,
umí vytvářet běžné sestavy a dovede upravovat výstupy pro jejich další distribuci.

Tabulka 45 Požadavky databázového modulu ECDL

Definice nároků na základní znalostní prvky principu fungování datové báze:
znát klíčové pojmy,
pochopit, co je databáze,
chápat rozdíl mezi daty a informacemi,
pochopit organizaci databáze a chápat pojmy tabulka, záznam a pole,
znát souvislost mezi praktickou aplikací databází a uživatelským pohledem.

Tabulka 46 Teoretické znalosti databázového myšlení

Definice nároků na znalosti v oblasti struktury databáze:
pochopit, že každá tabulka databáze zachycuje data pouze o jednom typu entity
pochopit, že každé pole tabulky by mělo obsahovat pouze jeden typ dat
pochopit, že obsah pole tabulky je spojen s datovým typem, jako je text, číslo, datum a čas
pochopit, že pole tabulky mají vlastnosti, jako je velikost pole, formát pole a výchozí hodnota
vědět, co je primární klíč tabulky
vědět, co je index, a pochopit, jak index umožňuje rychlejší přístup k datům

Tabulka 47 Znalosti v oblasti struktury databáze

Definice nároků na znalosti v oblasti principů relační integrity:
pochopit, že relační integrita omezuje výskyt redundantních dat v databázi,
vědět, že relace určuje vazbu mezi primárními klíči tabulek,
pochopit principy dodržení správnosti vazeb mezi tabulkami (referenční integrity).

Tabulka 48 Znalosti v oblasti relace

Definice nároků na znalosti a dovednosti v kompetenční oblasti databázového specialisty:
vědět, že profesionální databáze jsou navrhovány a vytvářeny databázovými specialisty,
vědět, že zadávání dat, údržba dat a získávání informací jsou prováděny uživateli,
vědět, že správce databáze poskytuje oprávnění pro přístup k určitým datům v databázi,
vědět, že správce databáze je odpovědný za správné fungování databáze po její havárii.

Tabulka 49 Znalosti v oblasti obsluhy databáze

Definice nároků na kompetence v oblasti použití databázové aplikace:
znalost základních principů práce s databázemi,
schopnost spustit a ukončit databázovou aplikaci,

otevřít a zavřít databázi,
vytvořit novou databázi a uložit ji na konkrétní místo souborového systému,
zobrazit a skrýt panely nástrojů, obnovit a minimalizovat lištu panelu nástrojů,
používat dostupné funkce programové nápovědy.

Tabulka 50 Uživatelské dovednosti v oblasti použití databázové aplikace

Definice nároků na kompetence v oblasti běžných pracovních úkolů s databází:
otevírat, ukládat a zavírat tabulky, dotazy, formuláře a sestavy,
přepínat mezi režimy zobrazení návrhu tabulky, dotazu, formuláře a sestavy,
odstranit tabulku, dotaz, formulář a sestavu,
schopnost navigovat mezi záznamy v tabulce, dotazu a formuláři,
specifikovat kritérium pro uspořádání záznamů v tabulce, formuláři a sestavě.

Tabulka 51 Běžné dovednosti v oblasti použití databázové aplikace

Definice kompetencí v oblasti práce se záznamy v tabulce:
přidávat a odstraňovat záznamy v tabulce,
vkládat, upravovat a mazat data v záznamech.

Tabulka 52 Dovednosti pro práci se seznamy v tabulce

Definice kompetencí v oblasti návrhu vnitřního uspořádání tabulky a konfigurace zobrazení:
vytvořit a pojmenovat tabulku, specifikovat pole a jejich datové typy,
nastavit rozšířené vlastnosti, velikost, zobrazovaný formát a výchozí hodnotu pole,
definovat pravidla pro verifikaci vkládaných formátů údajů,
vyhodnotit následky modifikace datového typu a vlastnosti sloupce tabulky,
nastavit pole jako primární klíč databáze,
schopnost konfigurovat indexaci pole (s povolením a se zakázáním duplicit),
rozšířit existující tabulku o další sloupec,
měnit vlastnosti zobrazení sloupce v tabulce a znalost vlivu změny zobrazení.

Tabulka 53 Dovednosti pro návrh tabulky

Specifikace kompetencí v oblasti hlavních operací pro získání informací:
používat vyhledávací příkaz pro nalezení určitého údaje v tabulce,
používat filtr v tabulkách a formulářích,
odstranit nastavený filtr z tabulky a formuláře.

Tabulka 54 Dovednosti pro získávání informací

Specifikace kompetencí pro získávání informací formulací dotazů:
vědět, že dotazy jsou používány pro vyhledání a analýzu dat,
vytvořit a pojmenovat dotaz nad jednou tabulkou s použitím určitých kritérií pro vyhledávání,
vytvořit a pojmenovat dotaz nad dvěma tabulkami s použitím určitých kritérií pro vyhledávání,
zadávat kritéria dotazu s využitím jednoho nebo více z následujících operátorů: = (rovno), <> (není rovno), < (menší než), <= (menší než nebo rovno), > (větší než), >= (větší než nebo rovno),
zadávat kritéria dotazu s využitím jednoho nebo více z následujících logických operátorů: A (AND), NEBO (OR) a NE (NOT),
využívat zástupní znaky v dotazech,* nebo %, ? nebo, _
přidávat, upravovat a odstraňovat kritéria v dotazech,
přidávat, odstraňovat, přesouvat, skrývat a zobrazovat pole v dotazech,
spustit dotaz.

Tabulka 55 Dovednosti pro získávání informací dotazy

Specifikace kompetencí v oblasti návrhu formuláře:
pochopit, že formulář se používá pro zobrazení a údržbu záznamů,
vytvořit a pojmenovat formulář,
používat formulář pro vkládání nových záznamů,
používat formulář pro odstraňování záznamů,
používat formulář pro vkládání, úpravu a mazání dat v záznamech,
vkládat a upravovat text v záhlaví a zápatí formuláře.

Tabulka 56 Dovednosti pro získávání informací formulářem

Specifikace kompetencí pro vytváření databázových sestav:
znát pracovní postup pro vytváření sestavy, export dat,
pochopit, že primární účel sestavy je tisk vybraných informací z tabulky nebo dotazu,
vytvoření a pojmenování sestavy podle struktury tabulky, dotazu,
změna uspořádání datových polí a záhlaví v návrhu sestavy,
schopnost použít agregační funkce v souhrnné sestavě podle kritéria seskupení,
vkládat a upravovat text v záhlaví a zápatí sestavy,
schopnost přizpůsobit výstup konkrétnímu požadavku na strukturu údajů.

Tabulka 57 Dovednosti pro tvorbu výstupů z databáze

Specifikace kompetencí v oblasti tisku:
změnit vlastnosti tisku databázového objektu tisk na výšku, na šířku, formát papíru,
tisknout stránku, vybraný záznam nebo záznamy a tisknout celou tabulku,
tisknout všechny záznamy a určité stránky s využitím formuláře,
tisknout výsledek dotazu,
tisknout určitou stránku nebo stránky sestavy a tisknout celou sestavu.

Tabulka 58 Dovednosti pro tisk výstupů z databáze