EKONOMICKÁ FAKULTA TUL

Bakalářská práce

SQL jako nástroj pro sledování ekonomiky v online hře

Studijní program: B6209 Systémové inženýrství a informatika
Studijní obor: Manažerská informatika

Autor práce: **Jan Pajer**
Vedoucí práce: Ing. David Kubát, Ph.D., Ing.Paed.IGIP
Katedra informatiky

Liberec 2023

EKONOMICKÁ FAKULTA TUL

Zadání bakalářské práce

SQL jako nástroj pro sledování ekonomiky v online hře

Jméno a příjmení:	**Jan Pajer**
Osobní číslo:	E19000229
Studijní program:	B6209 Systémové inženýrství a informatika
Studijní obor:	Manažerská informatika
Zadávající katedra:	Katedra informatiky
Akademický rok:	2021/2022

Zásady pro vypracování:

1. Vymezení základních pojmů.
2. Analýza současného stavu databáze.
3. Návrh vhodného řešení pro danou společnost.
4. Aplikace navrženého řešení.
5. Zhodnocení a závěr.

Rozsah grafických prací:
Rozsah pracovní zprávy: 30 normostran
Forma zpracování práce: tištěná/elektronická
Jazyk práce: čeština

Seznam odborné literatury:

- LIOY, Kevin, 2019. MySQL: SQL Database Programming for Beginners. Independently published. ISBN 978-1688905818.
- LAURENČÍK, Marek, 2018. SQL: podrobný průvodce uživatele. Praha: Grada Publishing. ISBN 978-80-271-0774-2.
- VYSTAVĚL, Radek, 2021. Databáze a SQL pro začátečníky. Ondřejov: Radek Vystavěl. ISBN 978-80-908144-0-0..
- PROQUEST, 2021. *Databáze článků ProQuest* [online]. Ann Arbor, MI, USA: ProQuest. [cit. 2021-09-26]. Dostupné z: http://knihovna.tul.cz/

Konzultant: Ing. Jiří Veselka - Vedoucí IT oddělení

Vedoucí práce: Ing. David Kubát, Ph.D., Ing.Paed.IGIP
Katedra informatiky

Datum zadání práce: 1. listopadu 2021
Předpokládaný termín odevzdání: 31. srpna 2023

L.S.

doc. Ing. Aleš Kocourek, Ph.D.
děkan

Ing. Petr Weinlich, Ph.D.
vedoucí katedry

V Liberci dne 1. listopadu 2021

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

14. června 2023 Jan Pajer

SQL jako nástroj pro sledování ekonomiky v online hře

Anotace

Cílem této bakalářské práce je navrhnout a implementovat nástroj za pomoci SQL, který usnadní sledování a řízení ekonomiky v online hře Ekura, kterou vyvíjí společnost Sindicate Entertainment s.r.o. Práce je rozdělena na část teoretickou a část praktickou. Teoretická část se zabývá vymezením základních pojmů z oblastí počítačových her, databází a jazyka SQL, MySQL a PHP, které jsou nezbytné pro porozumění s manipulací s herními daty. Praktická část se zaměřuje na aplikaci těchto konceptů a na vývoj nástroje pro správu ekonomiky online hry. Pro práci s databází, která je v MySQL je použit nástroj Navicat, který je také použit pro vytvoření grafu celkové inflace ve hře. Navrhované a implementované řešení je následně vyhodnocené.

Klíčová slova

Databáze, databázové systémy, MMORPG, MySQL, Navicat, online hry, PHP, SQL

SQL as a tool for tracking economy in an online game

Annotation

The aim of this bachelor's thesis is to design and implement a tool using SQL that will facilitate the monitoring and management of the economy in the online game Ekura, developer by Sindicate Entertainment s.r.o. The thesis is divided into a theoretical and a practical part. The teoretical part is focused on defining the fundamental concepts from the fields of computer fames, databases and the SQL, MySQL and PHP languages, which are necessary for understanding and manipulating game data. The practical part is dedicated to applying these concepts and developing a tool for managing the online game's economy. For to work with the database, which is in MySQL, the Navicat tool is used, which is also used to create a graph of total inflation in the game. The proposed and implemented solution is subsequently evaluated.

Key Words

Database, database systems, MMORPG, MySQL, Navicat, online games, PHP, SQL

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce, panu Ing. Davidu Kubátovi Ph.D, Ing.Paed.IGIP, za trpělivost a všechna doporučení v průběhu psaní této práce. Dále bych chtěl poděkovat kolegům ze společnosti Sindicate Entertainment s.r.o. za skvělou spolupráci a podporu.

Obsah

Seznam zkratek

ACID	Atomicity, Consistency, Isolation, Durability
CSS	Cascading Style Sheets
DBS	Database System
DDL	Data Definition Language
DML	Data Modification Language
FK	Foreign key
HTML	HyperText Markup Language
ID	Identification
MMORPG	Massivelymultiplayer online role playing game
MySQL	My Structured Query Language
NPC	Non-Player Character
PHP	Hypertext Preprocessor
PK	Primary key
RPG	Role-playing game
SŘBD	Systém řízení báze dat
SQL	Structured Query Language

Seznam obrázků

Úvod

Bakalářská práce se zabývá využitím jazyka SQL jako nástroje, pomocí kterého je možné sledovat ekonomiku online hry. Práce je rozdělena do dvou částí. První část je teoretická, vymezíme si v ní základní pojmy z oblasti počítačových her, databázových systémů, databázovou strukturou a jazyka SQL.

Praktická část má za cíl za pomoci SQL vytvořit nástroj, který nám bude pomáhat sledovat ekonomiku v online hře Ekura. Analyzujeme současný stav databáze, následně bude navrženo řešení pro sledování ekonomiky online hry pro společnost Sindicate Entertainment s.r.o., která provozuje již zmíněnou MMORPG hru Ekura. Poté aplikujeme navržené řešení a v konečném důsledku i zhodnotíme vytvořený nástroj.

Hry jsou silnou součástí našeho života všech lidí. Téměř každý si zahrál někdy nějakou hru. Nicméně jak jde čas, tak přecházíme od běžných her k těm počítačovým, kde máme na výběr širokou škálu her, které nás zabaví na dlouhé hodiny, aniž by bylo třeba se kamkoliv vydat. Jednoduše stačí mít elektrické případně i internetové připojení a nějaký počítač, na kterém je možné si hru spustit.

MMORPG je jedním z nejoblíbenějších typů her, kdy hrají hráči mezi sebou, zlepšují se, koordinují či zápasí. Avšak stále se v těchto hrách zlepšují, budují si svého hrdinu, jako by to byli oni sami jen v jiném alternativním světě. Mohou tak v jistém slova smyslu žít jiným životem.

Ve světě online her je třeba udržovat vlastní ekonomiku, aby se hra nezhroutila. V případě, že by došlo k výraznému výkyvu ekonomiky v online hře a hráči by měli určitého herního zboží mnoho či naopak příliš málo, mohlo by je to omrzet a přejít ke konkurenci. Zároveň je nutné sledovat tyto data pro případ přidávání nového herního obsahu do hry, aby se mohla ekonomika balancovat, neboť v případě, kdy hráči potřebují pro rozvoj svých herních postav předměty, kterých na hře je příliš málo, staví je to do pozice, že další rozvoj by mohl být přehnaně příliš časově náročný na úkor zábavy. Hráči by mohli nový herní obsah zavrhnout a nevyužívat jej, což by znamenalo, že desítky až stovky hodin vývoje by nemělo patřičný účinek. V opačném případě, kdy by nový herní obsah umožnil příliš snadné vydělávání v dané hře, by se

mohla prohloubit propast mezi novými a stálými hráči, kdy noví hráči by neměli téměř žádnou šanci se vyrovnat hráčům, kteří danou hru hrají výrazně déle.

Z tohoto důvodu je důležité, aby firmy provozující online hry měla co nejvíce dat v databázi a mohla je zkoumat a vyhodnocovat. Bez těchto dat není možné, aby ekonomika v online hře byla stabilní a hráči v ní spokojení.

1 Počítačové hry

V dnešní době jsou počítačové hry součástí našich životů. Poskytují lidem nespočetné množství zábavy napříč všemi věkovými kategoriemi. V počítačových hrách mohou hráči žít svůj druhý život a být kýmkoliv chtějí být, neboť je lze označit jako alternativní model skutečného světa, který má jiná pravidla. (Basler J., 2016)

Historie počítačových her začala zhruba před šedesáti lety. Jak šel vývoj kupředu, začaly se objevovat různé herní konzole, které následoval i osobní počítač, jenž mohl být využívaný v domácnosti, díky čemuž se rozvoj her výrazně zrychlil. Z počátku se jednalo spíše o hry, jež snažily napodobit sport nebo deskové hry v elektronické podobě. (Tišňovský P., 2011)

1.1 Dělení počítačových her

Počítačové hry je možné dělit dle různých kritérií, přičemž základní kritérium pro dělení je počet hráčů. V případě, že v dané hře hraje hráč pouze sám, pak se takové hře říká singleplayer. Pokud však v dané hře je možné, aby bylo připojeno více hráčů zároveň, pak je řeč o multiplayer, v multiplayerové hře hráči mohou spolupracovat nebo soupeřit mezi sebou. Další možností rozdělení jsou online počítačové hry, kdy je zapotřebí přístupu k internetu, aby hráči mohli tuto hru hrát a offline počítačové hry, v tomto případě není potřeba internetového připojení. (Basler J., 2016)

Nejčastěji se však počítačové hry dělí dle žánru. Přičemž jedním z nejoblíbenějších žánrů jsou RPG hry. Jedná se o žánr hry, který spočítá ve výběru hrdiny, za nějž hráč hraje, postupně svého hrdinu vylepšuje a zlepšuje jeho úroveň, případně plní určité herní úkoly. Do tohoto žánru patří také MMORPG, jedná se o RPG hru, která se hraje na internetu online. Hráči z celého světa mají možnost si tuto hru stáhnout a pokud mají internetové připojení, tak se do ní přihlásit a se svými spoluhráči v dané hře komunikovat, obchodovat či spolupracovat. (Basler J., 2016)

2 Databáze

Databáze je velké uložiště dat, kde je třeba udržovat určitou strukturu a sledovat vzájemné vztahy mezi daty, díky tomu se odlišuje databáze od uložiště, jaké jsou na počítačích, kdy může být velké množství fotek na disku, avšak i v případě, že vlastník těchto fotek se snaží mít pořádek v těchto fotkách, tak si složky roztřídí maximálně do složek. Kdežto v databázi je možné mít všechna možná data o podnikových datech, kdy jsou zaznamenávány i provázanosti. Mnoho artiklů zboží, který podnik prodává, prochází různým postupem, od příjmu zboží po různé schvalovací procesy až k prodeji. Databáze jsou tvořeny způsobem, který umožňuje, aby co nejvíce těchto dat bylo možné uložit a posléze opět dohledat. (Vystavěl R., 2021)

Další možnou definicí je ta, že databáze je uspořádaná soustava dat, přičemž zároveň dle této definice to znamená, že nezáleží, zda jsou data uchovávána v papírové či elektronické podobě. Jelikož je třeba, aby data bylo možné snadno upravovat, vyhledávat v nich či je odstraňovat, pak je jasné, že klasická papírová podoba, kdy jsou data zapsána v papírech bez jakéhokoli uspořádání, nesplňují. Kdežto data zapsána v papírových kartotékách lze prohlásit za předchůdce databází, jelikož v nich uspořádaná data jsou. (Laurenčík M., 2018)

Před vytvořením databáze je třeba jí vhodně navrhnout. Data v tabulkách by neměla obsahovat duplicity. Také je třeba, aby data byla rozdělena do tabulek, které budou logicky rozdělené a propojené za pomoci primárních klíčů, aby byla jistota, že data v databázi budou jednoznačně rozeznána a tedy, nebude více záznamů, které by byly stejně pojmenovány. Kdyby bylo více záznamů v databázi, které by měly však jiné hodnoty, nebylo by možné rozlišit, který záznam, k čemu patří, což vede k nepřehlednosti databáze. (Lioye K, 2019)

2.1 Uživatelé databázových systémů

Rozlišuje se několik typů uživatelů, kteří přistupují k DBS neboli databázovému systému. Prvním z těchto typů jsou správci databáze, což bývá skupina lidí, která se stará o databázi a má k ní plný přístup. Spravují databázi, zajišťuji přístupy do

databáze dalším skupinám na základě požadavků jedinců a jejich relevance. (Pokorný et al., 2020)

Mezi další skupiny uživatelů patří programátoři, „koncoví uživatelé“ a „příležitostní uživatelé“. Programátoři jsou zdatní v programovacích jazycích, a tedy většinou dobře ovládají například programovací jazyk SQL, jehož pomocí pracují s daty v databázi. Koncoví uživatelé většinou také ovládají nějaký z dotazovacích jazyků a jsou schopni si nějaké vlastní dotazy pro práci s databází sepsat. (Pokorný et al., 2020)

Poslední a největší skupinou jsou příležitostní uživatelé. Tito uživatelé již s databází nepracují jako ostatní skupiny. Většinou mají přístup do některé aplikace, kde mají dotazy již předpřipravené a oni je pouze doplňují, případně vybírají z nabídky. (Pokorný et al., 2020)

2.2 Historie a typy databázových modelů

První databázový model je model hierarchický model, který má hierarchickou strukturu, jež má vztah, kdy potomek má jednoho rodiče a naopak. Tento model však dlouho nevydržel, neboť nebyl příliš praktický. Rychle jej tedy nahradil model síťový, který doplňuje hierarchický model. Hlavní výhodou síťového modelu oproti hierarchickému byl ten, že umožňuje vztah typu „více ku více“, což znamená, že jedna entita může mít více nadřazených záznamů neboli rodičů. (Laurenčík M., 2018)

Roku 1970 následovala koncepce relační databáze, jež tvoří základ dnes používaných databázových systémů. V relační databázi se data uspořádávají do řádků a sloupců, přičemž se data ukládají v tabulkách. Hodnoty sloupců mohou být pro různé tabulky společné, díky čemuž je možné vyjádřit vzájemný vztah mezi datovými tabulkami, které se používají i v dnešní době. (Laurenčík M., 2018)

Relační model databáze poskytuje konzistenci údajů, což je jeho primární výhodou, neboť důležitou součástí relačních databází je spolehlivé zajištění zpracování databázových transakcí, k tomu nám dopomáhá sada vlastností se zkratkou ACID. (Ian, 2016)

Firma IBM vytvořila dotazovací jazyk SQL, který slouží pro práci s relačními databázemi, jež jako první začala využívat firma Oracle, následný rozvoj relačních databází pokračoval díky nástupu internetu. (Laurenčík M., 2018)

2.3 Databázový systém

Pojem databázový systém označuje program neboli software, který poskytuje nástroje pro práci s databází. Umožňuje upravovat data, vyhledávat či je třeba i mazat. Je kladen důraz na výkon a také na jeho optimalizaci. (Laurenčík M., 2018)

2.4 Systém řízení báze dat

Systém řízení báze dat neboli SŘBD je skupina programů, která zajišťuje práci s databází. SŘBD společně s databází pak tvoří databázový systém, tedy DBS. SŘBD zahrnuje například techniky transakčního zpracování, řídí souběžný přístup více uživatelů k databázi, ochranu dat či zotavování z chyb. (Pokorný et al., 2020)

2.5 Transakce

Pod pojmem transakce se označuje posloupnost akcí, se kterou se zachází jako s jedním celkem. Může se jednat o akce jako je zápis nebo čtení. (Pokorný et al., 2020)

Příklad transakce může být převod peněz v bance mezi vlastními účty, tedy z jednoho účtu na druhý. V takovém případě se jedná o více operací, kdy se nejdříve musí strhnout peníze z jednoho účtu, načež se vytvoří a uloží záznam o převodu peněz, a nakonec se peníze připíší na druhý účet, kam jsme je nechali v bance zaslat. Byť se jedná o více operací, označuje se to jako jedna transakce.

2.6 ACID

Jedná se o standardní sadu vlastností, která se zabývá tím, jak se databáze zotavuje v případě, kdy došlo k nějakému selhání při zpracování transakce. Zkrátka ACID se skládá z následujících částí:

- Atomicity – buď vše, anebo nic. Pokud selže jedna část transakce, pak selže celá transakce, což nám zaručí, že buď bude každá transakce úspěšná anebo neúspěšná jako celek.
- Consistency – díky konzistenci je zaručené, že všechna data budou platit dle všech pravidel, která jsou definována, a to včetně omezení, kaskád a spouštěčů, jež se v databázi použily.
- Isolation – izolace zaručuje, že každá transakce bude spouštěna samostatně a není jakkoliv ovlivňována jinou transakcí. Z toho vyplývá, že transakce nemůže číst z žádné jiné, která ještě nebyla dokončena.
- Durability – durabilita neboli trvanlivost znamená, že ihned, co úspěšně skončí transakce, tak již zůstane v systému, a to i v případě, kdyby došlo k pádu systému bezprostředně po dokončení dané transakce. Není možné, aby systém sdělil, že transakce byla úspěšná, kdyby tomu tak nebylo. (Ian, 2016)

Lze se setkat s tím, že o těchto vlastnostech, jež tvoří základní principy transakčního zpracování hovoří jako o ACID testu transakcí. (Pokorný et al., 2020)

2.7 Databázové tabulky

Drtivá většina serverů má data uložená v datových tabulkách. Práce s tabulkami sestává ze dvou kroků. Tím prvním je nadefinování toho, jak tabulka bude vypadat a druhým krokem je následné plnění této tabulky. (Vystavěl R., 2021)

Základ tabulek tvoří řádky a sloupce. Každá tabulka by poté měla mít sloupec, kde je záznam, který je jednoznačně charakterizovaný. To označujeme jako PK neboli Primary key či česky primární klíč. Vazby mezi tabulkami se poté zajišťují za pomoci FK, což je Foreign key nebo česky cizí klíč. (Laurenčík M., 2018)

2.7.1 Index

Indexy slouží k rychlejšímu přístupu k datům z tabulky či pohledu. Indexů může být v tabulce více nemusí být pouze jeden. Jedním speciálním typem indexu je primární klíč, který slouží k jednoznačnému identifikování záznamu v tabulce. (Roth J., 2019)

2.7.2 Charakteristika primárního klíče

Jak již bylo zmíněno, tabulka by měla mít sloupec, díky kterému se jednoznačně určí záznamy v řádcích. V tomto sloupci jsou hodnoty, které jsou unikátní a nehrozí tedy jejich případné duplikování, které by znemožnilo jedinečnou identifikaci. Nejobvyklejším pojmenováním tohoto sloupce je „ID“. (Vystavěl R., 2021)

ID se využívá v případech, kdy tabulka nemá svůj přirozený primární klíč. Ten mohou mít třeba osoby. Při narození jedinec obdrží rodné číslo, což je jeho unikátní primární klíč. U produktů to mohou být pro změnu čísla produktů. Pokud je to tedy možné, doporučuje se využívat právě přirozené primární klíče. (Harrington J., 2016)

V praxi se však často vytváří primární klíče i u produktů či osob, které rodné číslo mají. Může nastat situace, kdy dojde k chybě a třeba dvě osoby mají jedno rodné číslo, v takovém případě by došlo ke zdvojení a nebylo by možné jednoznačně určit, k čemu daný záznam patří.

Vytváření tabulek bez primárního klíče se nedoporučuje, neboť primární klíč je nezbytný pro vazby mezi tabulkami. S primárním klíčem je zpracování tabulky také efektivnější než, kdyby tabulka žádný neměla. Vytvoření primárního klíče je možné i za pomoci více sloupců v tabulce. Nemusí se tedy jednat pouze o číslovanou řadu počínaje číslem jedna, ale i klidně spojením třeba čísla faktury a datumu. (Laurenčík M., 2018)

2.7.3 Funkce cizího klíče

Vazby mezi tabulkami zajišťuje cizí klíč. Cizí klíč je v jedné tabulce definovaný, odkazuje zároveň na sloupce v jiné tabulce, kde takový sloupec musí být buď primárním klíčem nebo alespoň splňovat podmínku, že takový sloupec obsahuje

pouze unikátní záznamy a nehrozí tedy jejich zdvojení. Tyto sloupce musí být stejného datového typu. Cizí klíče nesou informace o tom, na jaké tabulce je tabulka závislá. (Stain S., 2017)

Tabulky také mohou mít cizí klíč samy na sebe. Jako příklad se často uvádí například tabulka se zaměstnanci, která obsahuje i sloupec „šéf", jenž má vlastní identifikátor, avšak i šéf je zaměstnanec dané firmy. Cizí klíč pak odkazuje na sloupec ve stejné tabulce. (Harrington J., 2016)

2.8 Relace

Relacemi jsou označovány vazby mezi tabulkami. Relace jsou základem aktuálně nejpoužívanějšího typu databází, a tedy relační databáze. Relace jsou velmi využívány jazykem SQL. (Laurenčík M., 2018)

Funkční relace vyžaduje, aby spojení mezi tabulkami bylo jednoznačné. Pokud nebude alespoň v jedné tabulce sloupec primárním klíčem, nebude relace funkční. Další podmínkou je, aby v obou tabulkách byly údaje stejného datového typu. (Laurenčík M., 2018)

Dle Laurenčíka (2018) existují tři typy relací mezi tabulkami:

- Relace 1:1 – řádek v jedné tabulce odpovídá řádku v druhé tabulce. Spojení tabulky touto relací se v praxi příliš nevyužívá, jelikož se většinou zařazují všechny sloupce do jedné tabulky.
- Relace 1:N – zde se jedná o spojení tabulek, kde v záznam v první tabulce odpovídá žádnému, jednomu či více záznamům v tabulce druhé. Jedná se o v praxi nejpoužívanější relaci.
- Relace N:N – neboli „více ku více". Každý řádek v první tabulce může mít buď žádný, jeden či více záznamů v druhé tabulce a naopak.

Při tvorbě databáze je dobré ukládat relace společně s referenční integritou, aby byla zachována konzistence ukládaných dat do databáze. Doporučuje se používat restriktivní referenční integritu, jelikož je bezpečnější, co se týče chyb jedince, neboť v případě, že by se jedinec dopustil chyby, mohlo by to způsobit značné komplikace,

protože by mohl smazat i záznamy v tabulkách, která by smazat nechtěl. (Laurenčík M., 2018)

2.9 Referenční integrita

Referenční integrita je definována cizím klíčem. Jedná se o podmínky, díky kterým je možné udržovat vztahy mezi tabulkami. Základní podmínkou je, aby se v tabulce nevyskytla situace, kdy cizí klíč neodpovídá žádnému záznamu primárního klíče, ačkoliv opačně je to v pořádku a ničemu to nevadí. (Laurenčík M., 2018)

Omezující podmínky hlídají v tabulkách pole, které mají společné hodnoty. První omezující podmínky se označují jako restriktivní a druhá jako kaskádová. V případě restriktivních pravidel referenční integrity nelze smazat nadřízený záznam v případě, že na něj odkazuje podřízený záznam. Kaskádová referenční integrity umožní změny v tzv. kaskádě, pokud například smažeme nadřazený záznam, smaže se i podřízený záznam v druhé tabulce. (Laurenčník M., 2018)

2.10 Hodnota NULL v databázi

Zaměňovat hodnotu NULL v databázi s hodnotou „0“ je špatně, ačkoliv je k tomu sváděno, hodnota NULL znamená, že se neví, co na daném políčku má být. Tedy pokud v nějakém řádku je hodnota rovna NULL, pak hodnota tohoto políčka není žádná. Při práci s daty, v případě, kdy by někdo zkoušel položit podmínku pro filtrování „NULL = NULL“ je výsledek opět NULL. Neboť se neví, zda je tato podmínka pravdivá či nepravdivá. (Laurenčík M., 2018)

3 SQL

SQL je zkratka Structured Query Language neboli Strukturovaný dotazovací jazyk. Tímto jazykem se komunikuje s databázovým serverem. V případě, že aplikační část serveru nebo i člověk, většinou správce databázi či programátor, potřebuje komunikovat s databázovým serverem, pak svůj příkaz či dotaz formuluje právě jazyce SQL. (Vystavěl R., 2021)

SQL je neprocedurální jazyk, což znamená, že na rozdíl od procedurálních jazyků se v něm popisuje, co se od databáze požaduje, nikoliv, jak se to má provádět. V SQL je možné definovat data a provádět aktualizace. Lze v něm nadefinovat i přístupová práva do databáze. (Pokorný et al., 2020)

V SQL je možné definovat i datové struktury, které umožňují rychlejší přístup k řádkům tabulek během zpracovávání požadavku neboli indexy. Jelikož indexy nepatří do logického datového modelu, jenž poskytuje SQL, pak není jejich definice standardizována. Indexy či tabulky je možné průběžně definovat, modifikovat či mazat. (Pokorný et al., 2020)

3.1 Definice dat

V SQL odpovídá deklarace dat dvourozměrné tabulce. Jména tabulek by měly být jednoznačná, stejně tak jména sloupců. Jméno tabulky se nedává jako je jméno sloupců, aby nedocházelo k problémům s tím spojených následně při práci s daty v tabulkách. (Pokorný et al., 2020)

Při definici by se měly dodržovat základní zásady. Pokud je jeden sloupec ve více tabulkách, je vhodné jej pojmenovat stejně, jména tabulek pojmenovat jako podstatná jména v množném čísle a pro změnu jména sloupců zase jako podstatná jména v jednotném čísle. (Pokorný et al., 2020)

3.2 Práce s tabulkami

Pro práci s tabulkami nám v SQL slouží tři příkazy. První je CREATE TABLE, jeho pomocí vytváříme tabulku. Slouží k definování polí v tabulce a zároveň i omezení těchto polí. Pole může být omezeno na nenulové hodnoty „NOT NULL“, v takovém případě musí pole obsahovat platná data. (Pokorný et al., 2020)

Dalším příkazem je příkaz ALTER TABLE, ten slouží pro změnu již vytvořené tabulky, pomocí tohoto příkazu je možné změnit sloupec, přidat další sloupec či nějaký sloupec pro změnu smazat. (Pokorný et al., 2020)

Posledním příkazem je příkaz DROP TABLE. Ten slouží pro odstranění tabulky z databáze. (Pokorný et al., 2020)

V případě příkazů ALTER TABLE a DROP TABLE záleží na referenční integritě databáze, která určuje, jak se příkaz zachová. Pokud je referenční integrita databáze nastavená na kaskádovou, tak když se použije příkaz DROP TABLE, tak dojde ke smazání všech objektů, jenž na danou tabulku odkazují. (Pokorný et al., 2020)

Práce s příkazy, které slouží pro definici dat, spadají do Data definition language tedy DDL příkazy. Pomocí těchto příkazů je možné taky vytvářet vazby mezi tabulkami a pracovat s indexy. (Laurenčík M., 2018)

3.3 Práce s daty

Pro práci s daty slouží v SQL čtyři příkazy. Nejdůležitější z nich je příkaz SELECT, který slouží k vybírání dat z tabulek v databázi. Pro práci s tímto příkazem je třeba znát alespoň část schéma databáze, neboť data mohou být ve více tabulkách pod jinými jmény. Znalost jmen tabulek a jejich atributů je tedy nedostačující. (Pokorný et al., 2020)

Příkaz SELECT se skládá z povinných klauzulí a nepovinných. Z těch povinných se jedná o klauzule právě SELECT a poté klauzule FROM, za níž se zadá název tabulky, ze které jsou data vybírána, což může být třeba i další SELECT, který vytvoří tabulku z jiné, a tak dokola. Nepovinnou nejčastěji používanou klauzulí je WHERE,

kterou definujeme, jaká data chceme vybírat. Příkaz SELECT má mnoho dalších klauzulí, které umožňují odstranění duplicit, seřazení tabulky či i různé výpočty. (Harrington J., 2016)

Pro vkládání řádků do vytvořené tabulky slouží příkaz INSERT. Na pořadí vkládaných dat pomocí příkazu INSERT nezáleží, neboť to neovlivní pořadí řádku v tabulce, do které jsou data vkládána. Pomocí příkazu INSERT spojeným s příkazem SELECT je možné do tabulky vložit výsledek dotazu, naplní se tedy hodnotami z jiné či více tabulek, která mohou být nějakým způsobem upravena či filtrována. (Pokorný et al., 2020)

Nejméně používanými příkazy jsou příkaz DELETE a UPDATE. Za pomoci příkazu DELETE se odstraňují záznamy z tabulek, nikoli však tabulka samotná. Příkaz UPDATE poté slouží k modifikaci záznamů v tabulkách. (Pokorný et al., 2020)

Příkazy, která slouží pro práci s daty jsou označovány jako DML příkazy tedy, Data Modification Laguage. (Laurenčík M., 2018)

3.4 Pohledy (view)

Pod pojmem „pohled“ si ve spojitosti s SQL lze představit virtuální tabulky v databázi. Tyto pohledy umožňují zpřehlednění práce s daty, a také sestavování dotazů. Hrají velkou roli v bezpečnosti dat, jelikož místo zpřístupnění nějaké určité tabulky se může vytvořit pohled, který bude s danou tabulkou totožný, avšak nemusí obsahovat určité sloupce, které obsahují citlivé informace. Takovýto pohled je poté možné zpřístupnit téměř komukoliv, aniž by hrozilo, že dotyčné osoby získají z této tabulky i data, která se uvádět nechtějí. K vytvoření pohledu se pak využívá příkaz CREATE VIEW. (Pokorný et al., 2020)

Jako příklad si lze uvézt tabulku s osobami, která obsahuje veškerá data o daných osobách, jako je jméno, adresa či rodné číslo. Pokud bude chtít někdo pracovat s daty z této tabulky a není vhodné, aby měl přístup i k rodným číslům, pak správce databáze jednoduše vytvoří pohled s tabulkou, která již nebude mít sloupec s rodným číslem a k té tabulce dotyčné osobě nastaví oprávnění ke čtení.

Pohledy mohou sloužit i k uložení některých složitějších dotazů v SQL, kdy jednoduše vytvoříte pohled právě z onoho SQL dotazu a ten v budoucnu lze znovu používat. Tento pohled poté může sloužit i jako zdroj dat namísto tabulky, není tedy třeba znovu vytvářet filtry na určité tabulky, když tyto odfiltrovaná data jsou již vytvořená v pohledu. Jen je třeba si dát pozor na pojmenování pohledu, to nesmí být stejné jako název jakékoliv již existující tabulky v databázi. (Laurenčík M., 2018)

3.5 Aktualizace pohledů

Ačkoliv aktualizace základních tabulek není nijak komplikovaná věc, tak aktualizace pohledů, je již komplikovanější, jelikož ne vždy se mohou použít operace sloužící k aktualizaci tabulky. Je tedy třeba aktualizovat tabulky, nad kterými jsou dané pohledy definované. (Pokorný et al., 2020)

Standard SQL definuje dva typy pohledů, ty, co jsou aktualizovatelné a ty, co nejsou. Jsou základní pravidla, která se u aktualizovatelných pohledů musí dodržovat. Je zakázané eliminování duplicitních záznamů za pomoci klauzule DISTINCT, také v klauzuli WHERE nesmí být použitý vnořený příkaz SELECT. Dále je zakázané použít klauzule GROUP BY či HAVING. Každý sloupec aktualizovatelného pohledu musí být zároveň sloupcem tabulky a může být vybrán pouze jednou. (Pokorný et al., 2020)

3.6 MySQL

MySQL je jeden z nejpopulárnějších open-source systém pro správu relačních databází založený na jazyku SQL. V MySQL se data ukládají v oddělených tabulkách, namísto jejich hromadění do jedné obrovské tabulky. Díky tomuto přístupu MySQL je dosaženo rychlejšího zpracovávání dat a flexibility. MySQL umožňuje přístup více uživatelům, přičemž dohlíží taky na to, aby přístup měli jen oprávnění uživatelé. (Welling et al., 2017)

MySQL je rychlý, může být využitý i pro práci s miliardami řádků. Klade také velký důraz na bezpečnost, jelikož podporuje šifrování dat a autentizaci uživatelů či umožňuje správu bezpečnostních práv. Integrace je možná s mnoha jazyky jako jsou PHP, Java, Python a další. (Welling et al., 2017)

Jedno z nejčastějších použití MySQL je se skriptovacím jazykem PHP. MySQL i jazyk PHP jsou kompatibilní a plně funkční na všech známých operačních systémech, dokonce i na těch méně známých. Hlavní výhodou PHP je jeho výkon. MySQL s použitím PHP je využíván k ukládání a získávání dat, která jsou zpracovávána pomocí PHP skriptů. (Welling et al., 2017)

Rozdíl mezi SQL a MySQL je takový, že SQL se používá pro interakci s databází, zatímco MySQL je konkrétní databázový systém, který tento jazyk využívá a přidává mnoho vlastních funkcí a rozšíření, které nejsou součástí standardu SQL.

4 Analýza současného stavu databáze

Společnost Sindicate Entertainment s.r.o., která provozuje online hru Ekura, pro níž je tento projekt zpracováván, má obsáhlou databázi, jež je postavena na technologii MySQL. Všechna tato data umožňují vhled do chování hráčů a ekonomiky online hry Ekura.

Data, jež jsou uchovávána na úrovni hráčů poskytují základní informace ohledně jmen postav, přihlašovacích údajů, e-mailů, IP adres či tzv. PCID. PCID je unikátní identifikátor, na jehož základě se v systému rozeznává jeden počítač od druhého. Jedná se o mix informací, jež jsou staženy ze zařízení při přihlášení do online hry skrze launcher. Jde o sériová čísla HW zařízení či licence Windows. Toto PCID je nejdostupnější informace k rozeznávání uživatelů od ostatních a nám může být velmi nápomocné při tvorbě nástrojů pro sledování ekonomiky. Jelikož však mnoho uživatelů má v domácnosti více zařízení, jeví se aktuálně jako lepší prostředek pro rozpoznávání uživatelů IP adresy.

Dalšími daty, která se týkají více ekonomických aspektů hry jsou data ohledně transakcí mezi hráči. Všechny možné obchody mezi hráči jsou zaznamenávány a ukládány do databáze pro případné použití. Databáze také obsahuje informace o množství herní měny, která je generována v rámci hry samotné.

Bohužel data v databázi se historicky ukládají momentálně pouze za aktuální a předchozí rok. Z tohoto důvodu jsou data v databázi od počátku roku 2022. Což může znepříjemňovat řešení problémů v případě dohledávání dat pro uživatele, kdy záznam o nějaké změně může být již smazán. Zároveň to komplikuje výsledné srovnávání inflace za delší časové období. S daty od počátku roku 2022 má smysl řešit inflaci dle měsíců, případně kvartální. Přesto by však bylo ideální, pokud by byla k dispozici data z dřívějších let, aby bylo možné porovnat stejné období s obdobími předchozími.

Ačkoliv společnost Sindicate Entertainment s.r.o. uchovává ve své databázi velké množství dat, prozatím s daty ve směru sledování ekonomiky v jejich online prostředí příliš nepracovala. Data sloužila převážně k řešení případných potíží hráčů, kdy docházelo ke ztrátám předmětů či řešení porušování podmínek použití. Nicméně je možné všechna tato data využít například ke sledování inflace měny hry a k jejímu

případnému vyrovnávání. Je také možné sledovat změny v ekonomii hry, které mohou souviset například s přidáváním nového herního obsahu a tím pomoci lépe balancovat ekonomiku herního prostředí.

4.1 Specifikace problému

Aktuálně největší problém spočívá v inflaci v online hře. Ceny všech předmětů ve hře dosahují vysokých částek a hromadění herní měny ve hře je jedním ze stále větších problémů hry. Do ekonomiky proudí stále více této měny a prostředky ve hře jsou aktuálně nedostačující k jejímu stahování ze hry. Ačkoliv ještě před pár lety bylo běžné, že v online hře byly vysoké transakce v řádu desítek miliard, nyní již částky za podobné předměty dosahují až k bilionu. Hráčům se herní měna na účtech hromadí a vytváří čím dál větší propast mezi stávajícími a novými hráči, kteří se je snaží dostihnout.

Součástí úprav v online hře je přidání nového herního obsahu, který má za cíl stahovat více herní měny z oběhu hry, zároveň rozptýlí herní měnu i mezi nováčky a pomůže snížit propast mezi hráči. Tento nový herní obsah bude vyžadovat od hráčů s vyšší úrovní obětování předmětů, které lze v online hře pořídit od jiných hráčů. Čímž bude docházet k přerozdělování herní měny od hráčů, kteří herní měnu hromadí, mezi hráče nové. Noví hráči tyto důležité předměty získávají v dřívějších fázích hry. Což bude mít za následek urychlení jejich startu. Přičemž v dřívějších fázích hry jsou již nyní úkony, které stahují herní měnu z oběhu hry, a tak bude docházet právě ke snižování množství herní měny, které je nyní v online hře skutečně velké množství.

4.2 Návrh vhodného řešení pro danou společnost

Vzhledem k novému hernímu obsahu, jež je v plánu přidat do hry a jehož prvotní účel je snižovat propast mezi novými a stávajícími hráči, je tedy třeba vytvořit nástroj, který bude sledovat inflaci v online hře pro vybrané předměty, které jsou vybrány vedením společnosti. ID těchto předmětů budou uložena v tabulce a bude tedy možné v budoucnu přidávat či ubírat předměty, které budou sledovány. Vzhledem k vývoji online her není možné vždy zcela přesně určit, zda některé předměty ze hry nezmizí

nebo zda se v průběhu let do hry nepřidají nové herní předměty, které by nahradily stávající.

Poté bude třeba vytvořit skript, který bude procházet prodeje předmětů z této tabulky a aktualizovat průměrnou cenu. Aby nedocházelo k přílišnému ovlivňování ceny extrémními výkyvy, kdy hráči použijí náhodou na přesun herní měny mezi účty prodej nějakého z těchto předmětů, tak bude použita hodnota mediánu z těchto prodaných předmětů. Také, aby nedocházelo k dalšímu případnému zkreslování, je třeba sledovat počet prodejů za dané období, aby v případě, kdy by bylo velmi málo prodejů, nedocházelo k dalšímu případnému zkreslení.

Je třeba eliminovat prodeje předmětů mezi jedním hráčem. Stává se, že hráči používají tyto prodeje pro přesun herní měny mezi vlastními účty, což může vytvářet další extrémní hodnoty, jelikož nejsou prodávány za tržní cenu, nicméně za cenu, kterou hráč potřebuje přesunout z jednoho účtu na druhý, byť k tomuto slouží jiné prostředky ve hře, je tento prostředek stále velmi často využíván. Je tedy potřeba stanovit limity, které eliminují uskutečňování těchto transakcí, jež nic nevypovídají o hodnotě daného předmětu. Zde se nabízí právě limitace sledování prodejů předmětů na různé IP adresy.

Jelikož je nežádoucí přímo kopírovat inflaci měny hry a promítat ji do cen předmětů v plném rozsahu, bude navýšení cen limitováno o určitá procenta, aby nedocházelo k rychlému a enormnímu tlaku na hráče s vyšší úrovní, neboť všichni tito hráči u sebe nemají velké množství herní měny a je třeba, aby měli čas se přizpůsobit. V tomto ohledu bude výpočet nové měny přepočítán v rozmezí +- 10 % oproti předchozí ceně. Tímto způsobem se zabrání prudkému nárůstu cen, který by mohl znevýhodnit hráče s nižším množstvím herní měny a zároveň by to poskytlo hráčům dostatek času a prostoru k adaptaci na novou ekonomickou situaci ve hře.

Inflace je ekonomický termín, jenž popisuje růst celkového cenového indexu zboží a služeb v ekonomice za určité období. Data pro průběžné sledování cen předmětů budou v tabulce aktualizována každých 7 dní. Zároveň budou zaznamenávána do další tabulky, aby byly historické záznamy o těchto změnách. Celková inflace bude zaznamenávána jednou za 28 dní, vzhledem k týdenním intervalům dat pro její

výpočet, a také tomu, že data obsažena v databázi jsou pouze od roku 2022, tudíž větší období měření inflace nemá prozatím příliš smysl vypočítávat.

Inflace takto prodaných předmětů bude sloužit k pozdějšímu využití při výpočtu celkové inflace, jež bude sloužit jako celkový vhled do ekonomiky online hry, kdy bude vytvořen graf, pro lepší přehled, jak se mění inflace v průběhu času, což bude sloužit jako ukazatel ekonomiky hry v určitém časovém období a zároveň ukazatel případného vlivu nového herního obsahu na ekonomiku hry.

5 Aplikace navrženého řešení

Jelikož je databáze v MySQL, byl zvolen nástroj podporující práci s touto technologií. Tento program se nazývá Navicat. Jedná se o komplexní software pro správu a vývoj databází. Navicat také umožňuje tvorbu grafů a reportů z dat připojené databáze.

5.1 Příprava tabulek

V první řadě je třeba vytvořit tabulku, do níž budou vkládána ID předmětů, jejichž medián z prodejů bude zaznamenáván. ID předmětů je v online hře Ekura označováno jako „vnum“. Tabulka bude pojmenována jako „price_check“. Všechny hodnoty v této tabulce budou typu integer, neboť se jedná o celočíselné hodnoty. V online hře Ekura nejsou neceločíselné hodnoty. Sloupec „vnum“ je nastaven jako primární klíč, tedy všechny hodnoty v tomto sloupci musí být unikátní a nemohou být nulové. Sloupce s primárním klíčem se často používají jako identifikační sloupec pro záznamy v tabulce. Následující kód uvádí vytvoření této tabulky:

```
CREATE TABLE price_check (
    vnum INT PRIMARY KEY,
    price INT,
    full_price INT
);
```

Pomocí příkazu INSERT se do této tabulky vloží do sloupce „vnum“ ID předmětů, které byly vybrány jako vhodné pro sledování. Jelikož je lepší mít více informací, jedná se o co největší množství předmětů, které je v řádu desítek předmětů. Zbylé dva sloupce se ponechají s hodnotou „NULL“ neboť u nich bude třeba provést aktualizaci cen pomocí skriptu.
Příklad vložení několika ID předmětů do tabulky „price_check“:

```
INSERT INTO price_check (vnum)
VALUES (30019), (30025), (30030), (30031), (30044), (30047),
(30050), (30091);
```

Další tabulkou, kterou je třeba si připravit, je tabulka „price_history“, do níž budou ukládána právě historická data. Tato tabulka již bude mít více sloupců, převážně jde

o sloupec datum, který umožní lepší vhled do průběžných změn cen daných předmětů v určitém datu. Dalším důležitým sloupcem je sloupec „sells“. Do tohoto sloupce se ukládá informace o počtu prodaných předmětů v daném, sledovaném, období. Ve sloupci „price“ poté vidíme aktuální cenu v daný datum, kdežto ve sloupci „old_price“ je uvedená předchozí cena.

Skript pro vytvoření této tabulky:

```
CREATE TABLE price_history (
    datum DATE,
    vnum INT PRIMARY KEY,
    price INT,
    sells INT,
    summary INT,
    old_price INT
);
```

5.2 Zpracování a získání dat z databáze

Nyní je třeba tabulky naplnit. Jelikož záznamy s transakcemi mezi hráči se skládají vždy ze dvou řádků „SHOP_BUY“ a „SHOP_SELL“ jež jdou vždy přímo za sebou a ve stejném pořadí, je třeba napsat skript, který tyto záznamy načte a spáruje jako jeden záznam. Společným ukazatelem je sloupec „what“, který má shodnou hodnotu v těchto transakcích, bohužel se záznamy mohou opakovat.

Dále tyto záznamy obsahují sloupec IP. Pokud mají tyto dva záznamy společnou IP adresu, je třeba, aby byly ignorovány. Právě pro případ, že se jedná s největší pravděpodobností o jednoho hráče, který provádí transakci mezi svými účty.

Tento skript je třeba napsat v jazyce PHP, neboť v MySQL by to bylo náročné a méně přehledné, vzhledem k tomu, že je třeba použít složité poddotazy a self-join operace pro srovnání každého záznamu s dalšími záznamy v tabulce. MySQL je primárně jazyk pro manipulaci a dotazování dat, pro složitější logické či sekvenční operace je vhodnější využít právě plnohodnotný programovací jazyk, který má více možností pro kontrolu toku a manipulaci s daty.

Jako první bude skript obsahovat připojení k databázi. Vytvoří se objekty: $dbi_log a $dbi, které představují připojení k databázi. Tyto objekty se inicializují voláním metody „get()“.

Následuje vytvoření konstant „SELLS_THRESHOLD“ a „DAYS_TO_LOAD“. Konstanta SELLS_THRESHOLD obsahuje minimální počet prodejů předmětu. Konstanta DAYS_TO_LOAD poté počet dní, které budou zpětně načítány. Tyto hodnoty jsou aktuálně nastaveny na hodnotu 10 a je možné je libovolně měnit:

```
define('SELLS_THRESHOLD', 10);
define('DAYS_TO_LOAD', 10);
```

Poté se připraví tabulka „price_check“, která je v databázi „common“. Tato tabulka byla vytvořena a naplněna ID předměty ve sloupci „vnum“. Do proměnné $vnumQuery se uloží jednoduchý SELECT pro výpis dat z tabulky. Tento SELECT se následně provede pomocí metody query(). Výsledek provedeného dotazu se nakonec uloží do nově vytvořené proměnné $vnumResult:

```
$vnumQuery = "SELECT vnum, price FROM common.price_check";
$vnumResult = $dbi->query($vnumQuery);
```

Pro uložení seznamu „vnum“ je třeba vytvořit pole. V níže přiloženém kódu je podmínka IF, ve které se zkontroluje, zda výsledek dotazu obsahuje alespoň jeden řádek. Následující while cyklus následně postupně prochází každý řádek výsledku dotazu, k čemuž jí pomáhá metoda „fetch_assoc()“, která získává jednotlivé řádky z výsledku jako asociativní pole:

```
$vnumArray = array();
if ($vnumResult->num_rows > 0) {
    while ($row = $vnumResult->fetch_assoc()) {
        $vnumArray[$row['vnum']] = array(
            'price' => $row['price'],
            'sells' => 0,
            'summary' => 0
        );
    }
}
```

Nyní je třeba ověřit, že pole $vnumArray není prázdné. V případě, že toto pole je prázdné, tak je třeba ukončit program a nepokračuje se dále, neboť v takovém případně nemá smysl pokračovat. K této kontrole se použije opět podmínka IF a pro případné ukončení programu příkaz „exit“:

```
if (empty($vnumArray)) {
    echo "Neexistují žádná 'vnum' k použití jako filtr.";
    exit;
}
```

Pokud pole obsahuje data, je možné pokračovat. Připraví se seznam „vnum“ předmětů z pole $vnumArray do řetězce $vnumList v upravené podobě, v jednoduchých uvozovkách, které jsou oddělené čárkami mezi sebou pro další práci s daty: `$vnumList = "'" . implode("','", array_keys($vnumArray)) . "'";`

Dále se vytvoří prázdné pole, které bude sloužit pro ukládání spárovaných záznamů. Také se vytvoří nový objekt třídy DateTime, který získává aktuální datum a čas, tuto hodnotu bude však třeba posunout o jeden den zpětně:

```
$paired_records = array();
$currentDate = new DateTime();
$currentDate->modify('-1 day');
```

Data je třeba projít, k procházení dat slouží cyklus „for“. Jelikož jsou data v denních logách, jde o tabulky s pojmenováním ve složení „rok_měsíc_den“, je tedy třeba tyto tabulky „vypočítat“. Níže uvedený kód má v sobě dva vnořené for cykly s několika podmínkami IF. Provádí procházení dat z tabulek, pro které zjistí název pomocí výpočtu na základě datumu. Dále připraví SQL dotaz s podmínkou na sloupec „vnum“ pomocí před generovaného seznamu, který je uložen v proměnné $vnumList. Následně prochází všechny výsledky a kontroluje, zda se po sobě jdoucí záznamy shodují v hodnotách ve sloupcích „how“ a „what“. Také kontroluje, zda se neshoduje IP adresa účtů v danou chvíli provedené transakce, pokud by tomu tak bylo, byla by tato transakce ignorována. Pomocí funkce „explode()“ se rozdělí obsah na více částí. Následně je proveden výpočet hodnoty „cena“ jako podílu „price“ a „count“. Nakonec se vytvoří záznam a přidá se do pole „$paired_records[]“. Po skončení této části je upraven datum:

```
for ($i = 0; $i < DAYS_TO_LOAD; $i++) {
    $tableName = "log_" . $currentDate->format('Y') . "." .
$currentDate->format('Y_m_d');
    $sql = "SELECT * FROM $tableName WHERE how IN ('SHOP_BUY',
'SHOP_SELL') AND vnum IN ($vnumList)";
    $result = $dbi_log->query($sql);
    if ($result->num_rows > 0) {
        $results = $result->fetch_all(MYSQLI_ASSOC);
        for ($j = 0; $j < count($results) - 1; $j++) {
            if ($results[$j]['how'] == 'SHOP_BUY' && $results[$j +
1]['how'] == 'SHOP_SELL' &&
                $results[$j]['what'] == $results[$j + 1]['what']) {
                if ($results[$j]['ip'] == $results[$j + 1]['ip']) {
                    continue;
                } else {
                    $hintParts = explode('  ',
$results[$j]['hint']);
                    $count = $hintParts[count($hintParts) - 2];
                    $price = $hintParts[count($hintParts) - 3];
                    $cena = $price / $count;
                    $vnum = $results[$j]['vnum'];
                    $paired_records[] = array(
                        'record1' => $results[$j],
                        'record2' => $results[$j + 1],
                        'cena' => $cena,
                        'vnum' => $vnum,
                        'count' => $count
                    );
                }
                $j++;
            }
        }
    }
    $currentDate->modify('-1 day');
}
```

Nyní je k dispozici pole „$paired_records“ s páry záznamů, které splňují podmínky za všech 10 dní. Každý záznam obsahuje také cenu jako hodnotu podílu price a count. Následně se vyhledají shodné záznamy dle „vnum“ a pro každé „vnum“ se vypočítá medián. K tomu bude sloužit cyklus foreach:

```
$medianValues = array();
foreach ($paired_records as $record) {
    $vnum = $record['vnum'];
    $cena = $record['cena'];
    $count = $record['count'];

    if (!isset($medianValues[$vnum])) {
        $medianValues[$vnum] = array();
    }
    $vnumArray[$vnum]['sells'] += 1;
    $vnumArray[$vnum]['summary'] += $count;

    $medianValues[$vnum][] = $cena;
}
$medianResults = array();
foreach ($medianValues as $vnum => $values) {
    sort($values);
    $count = count($values);
    $middle = floor(($count - 1) / 2);

    if ($count % 2) {
        $median = $values[$middle];
    } else {
        $lower = $values[$middle];
        $upper = $values[$middle + 1];
        $median = ($lower + $upper) / 2;
    }

    $medianResults[$vnum] = $median;
}
```

Poté je třeba výstup formátovat ve formě html tabulky, za pomoci CSS stylů. Výsledky, které jsou v „$medianResults“ se budou dále zpracovávat pro další analýzu.

Pomocí foreach se prochází pole „$medianResults“ a ukládá výsledky do proměnných. Získané data rovnou zpracovává zaokrouhlením. Nakonec je vypočítávána nová cena každého předmětu s ohledem na omezení, která jsou definována aktuálně pro rozmezí + a – 10 %:

```
echo "<style>
    table {
        border-collapse: collapse;
    }
    th, td {
        border: 1px solid black;
        padding: 5px;
    }
</style>";

echo "<table>
    <tr>
        <th>VNUM</th>
        <th>Median</th>
    </tr>";
foreach ($medianResults as $vnum => $median) {
    $roundedMedian = round($median);
    $sells = $vnumArray[$vnum]['sells'];
    $previousPrice = $vnumArray[$vnum]['price'];
    $minPrice = $previousPrice * 0.9;
    $maxPrice = $previousPrice * 1.1;
    $newPrice = max($minPrice, min($maxPrice, $roundedMedian));
```

Nakonec se provede aktualizace dat sloupců „price“ a „full_price“ v tabulce „price_check“ s novými hodnotami, avšak pouze v případě, že je počet prodaných položek větší, než je hodnota konstanty „SELLS_THRESHOLD“. Zaokrouhlené mediány se uloží do tabulky „price_history“ v databázi „common“. A provede se generování informativního výstupu v rámci tabulky. Každý zpracovaný „vnum“ je vložen jako řádek tabulky s odpovídajícími hodnotami „vnum“ a „newPrice“, kde se pro zobrazení používá HTML kód v rámci elementů „<tr>“ a „<td>“, které definují řádky a buňky tabulky:

```
if ($sells > SELLS_THRESHOLD) {
```

```
        $updateQuery = "UPDATE common.price_check SET price =
'$newPrice', full_price = '$roundedMedian' WHERE vnum = " . $vnum;
        $dbi->query($updateQuery);
    }
    $insertQuery = "INSERT INTO common.price_history (datum, vnum,
price, sells, summary) VALUES (NOW(), '$vnum', '$roundedMedian',
'{$vnumArray[$vnum]['sells']}',
'{$vnumArray[$vnum]['summary']}')";
    $dbi->query($insertQuery);
    echo "<tr>
        <td style='text-align: center;'>{$vnum}</td>
        <td style='text-align: center;'>{$newPrice}</td>
    </tr>";
}
echo "</table>";
?>
```

Tento skript je vhodné pouštět jednou týdně, k tomu může sloužit tzv. Cron. Jedná se o časovací a plánovací systém, který se běžně využívá v operačních systémech Unix a Linux. Slouží k automatickému spouštění skriptů, programů či příkazů v určitém časovém intervalu nebo v určitý čas.

Cron funguje na základě tzv. tabulky „crontab“, jenž obsahuje naplánované úlohy neboli „joby“ a jejich časové plány. Tabulka se nachází v cestě „/etc/crontab“ a její formát vypadá následovně:

```
#minute hour mday month wday who command
```

Následující příkaz ukazuje příklad použití Cronu, kdy bude prováděn jednou týdně, každé pondělí, což se nastaví pomocí hodnoty „1“, ve 3 hodiny a 40 minut ráno, kdy * je zástupný znak pro „každý“ a root pro řízení přístupových práv. Příkaz „fetch“ pak slouží k načítání obsahu ze zadané adresy skriptu. Skript je Cronem následně spouštěn z umístění, které je uvedeno:

```
40  3  *  *  1  root  fetch https://server/adresa_skriptu
```

Cron je velmi užitečný nástroj pro automatizaci. Podporuje několik speciálních řetězců, které usnadňují plánování úloh. Jako příklad lze uvést „@reboot“, který spustí úlohu po každém restartu systému.

5.3 Úprava tabulek a aktualizace dat

Pro zjištění inflace měny jednotlivých předmětů jsou data připravena. Celková inflace bude vypočítána na základě změn cen jednotlivých předmětů, které jsou uvedeny v tabulce „price_check“, nicméně již bude třeba pracovat s tabulkou „price_history“, ve které se ukládají i historické informace o cenách, nikoli pouze aktuální.

Pro úpravu základní tabulky pro výpočet změn cen jednotlivých předmětů se vytvoří procedura. Procedura je seskupení logických kroků, příkazů a instrukcí. Hlavní výhodou procedur je jejich znovupoužitelnost, neboť procedura může být vyvolána z různých částí programu.

Vytvořená procedura má název „generate_alter_statements()“. V této proceduře se deklarují proměnné pomocí „DECLARE“. Vytvoří se proměnná „done“, což je logická proměnná pro kontrolu ukončení cyklu, dále proměnná „alter_cmd“, která slouží pro ukládání generovaných ALTER příkazů a nakonec proměnná „cur“. Jedná se o kurzor pro iteraci přes výsledky dotazu. Poté se provede dotaz SELECT, který slouží pro vytvoření seznamu ALTER příkazů, jenž budou sloužit k přidání sloupců „vnum_x“ do tabulky „price_table“. Každý ALTER příkaz se generuje pomocí funkce CONCAT a obsahuje název sloupce ve formátu „vnum_x“, přičemž „x“ označuje hodnotu získanou z výsledků poddotazu:

```
CREATE PROCEDURE generate_alter_statements()
BEGIN
    DECLARE done INT DEFAULT FALSE;
    DECLARE alter_cmd VARCHAR(1000);
    DECLARE cur CURSOR FOR
        SELECT CONCAT(
            'ALTER TABLE price_table ADD COLUMN vnum_',
            vnum,
            ' INT;'
        ) AS alter_statement
        FROM (
            SELECT DISTINCT vnum
            FROM price_history
        ) AS subquery;
```

V případě, že nejsou žádné další řádky v kurzoru nastaví se proměnná „done“ na hodnotu TRUE, což způsobí, že se cyklus ukončí. V tomto bloku kódu je prováděno čtení výsledků z kurzoru a postupně generovány a vykonávány dynamické SQL příkazy pro přidání sloupců do tabulky „price_table“ na základě hodnot uložených v proměnné „alter_cmd“:

```
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    OPEN cur;
    read_loop: LOOP
        FETCH cur INTO alter_cmd;
        IF done THEN
            LEAVE read_loop;
        END IF;
        SET @sql = alter_cmd;
        PREPARE stmt FROM @sql;
        EXECUTE stmt;
        DEALLOCATE PREPARE stmt;
    END LOOP;

    CLOSE cur;
END
```

Druhou procedurou je procedura „update_price_table()“, jenž slouží pro samotné naplnění dat. Cíl této procedury je aktualizovat hodnoty v tabulce „price_table“ ve sloupci „vnum_x“ na základě dat z tabulky „price_history“. Za předpokladu, že počet prodaných položek, jenž se ukládá ve sloupci „sells“ je větší než 9. Tato procedura je podobná předchozí proceduře, převážný rozdíl je ve funkci „CONCAT“, která obsahuje „UPDATE“, jenž v SQL slouží pro aktualizaci hodnot:

```
CREATE PROCEDURE update_price_table()
BEGIN
  DECLARE done INT DEFAULT FALSE;
  DECLARE _vnum INT;
  DECLARE _price INT;
  DECLARE _datum DATE;
```

```
  DECLARE cur CURSOR FOR SELECT vnum, price, datum FROM
price_history where sells > '9' ORDER BY datum;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
  OPEN cur;
  read_loop: LOOP
    FETCH cur INTO _vnum, _price, _datum;
    IF done THEN
      LEAVE read_loop;
    END IF;
    SET @sql = CONCAT('UPDATE price_table SET vnum_', _vnum, ' = ',
_price, ' WHERE datum >= ', '"', _datum, '"');
    PREPARE stmt FROM @sql;
    EXECUTE stmt;
    DEALLOCATE PREPARE stmt;
  END LOOP;
  CLOSE cur;
END
```

Nyní je již možné pomocí série příkazů vypočíst změny cen předmětů. Nejprve se v případě, že tabulka existuje, provede smazání tabulky „price_table“, to je užitečné při zamezení chyb při pokusu o vytvoření tabulky, která již existuje:

```
DROP TABLE IF EXISTS price_table;
```

Poté se znovu tabulka pro výpočty jednotlivých hodnot dle „vnum“ opět vytvoří. Při vytváření tabulky se použije pouze sloupec „datum“ s primárním klíčem nad tímto sloupcem:

```
CREATE TABLE price_table (
  datum DATE PRIMARY KEY
);
```

K již vytvořené tabulce se doplní unikátní datumy z tabulky „price_history“, aby bylo zajištěno, že jsou zaznamenány všechny relevantní data, aniž by musely být manuálně zadávány po jednom:

```
INSERT INTO price_table (datum)
SELECT DISTINCT datum
FROM price_history;
```

Pomocí příkazu „call“ se následně procedura zavolá neboli provede. Vyvolání těchto procedur se provede kód, který obsahují uvnitř jich samotných, díky tomu se přidají sloupce v tabulkách a naplní případně aktualizují data:

```
call generate_alter_statements();
call update_price_table();
```

5.4 Celková inflace měny ve hře

Všechna důležitá data pro výpočet celkové inflace již k dispozici jsou. Celková inflace bude sloužit jako vhled do ekonomiky online hry a je složená z dílčích změn cen předmětů vypočtených v dřívějších krocích. Tato hodnota bude uložená v procentech v další tabulce „price_inflation_history“. Tuto tabulku je třeba vytvořit a naplnit daty. Vzhledem k tomu, že se do tabulky budou zapisovat vypočtená data z jiné tabulky, bude třeba tabulku vždy před plněním smazat, vytvořit a až poté naplnit daty pomocí příkazu INSERT.

Příkaz INSERT se smíchá se SELECTEM, ve kterém bude proveden výpočet inflace za 28 dní z tabulky „price_history“. Data se přiřadí do sloupců „datum“, „vnum“ a „inflation“. K této operaci slouží následující kód:

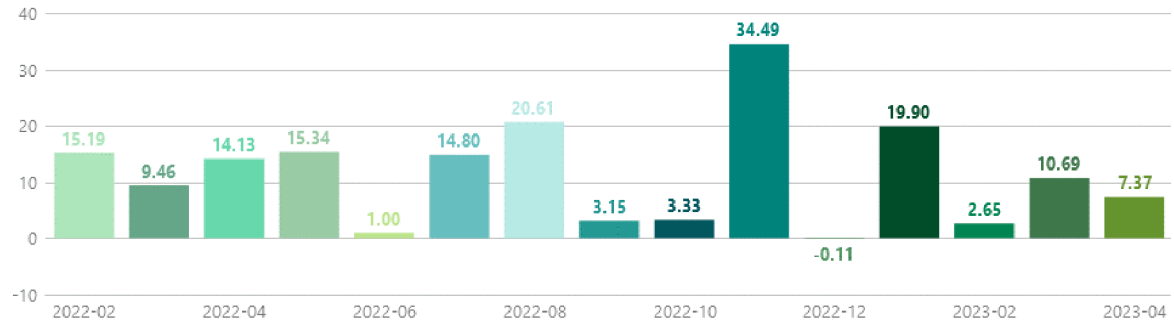
```
INSERT INTO price_inflation_history (datum, vnum, inflation)
SELECT p1.datum, p1.vnum, (p1.price - p2.price) / p2.price * 100 AS
inflation
FROM price_history p1
JOIN price_history p2 ON p1.vnum = p2.vnum
WHERE DATEDIFF(p1.datum, p2.datum) = 28;
```

5.5 Zobrazení celkové inflace měny ve sloupcovém grafu

Pro zobrazení celkové inflace měny hry, dle měsíců, lze použít program Navicat. V tomto programu je možné připojit data z databáze a následně je vložit do grafu. V Navicatu je také možné vytvářet dashboardy, které slouží k přehledné vizualizaci a sledování důležitých informací, dat a ukazatelů na jednom centrálním místě.

Vytváření grafu v Navicatu není složité. Přímý postup, jak se dostat do části programu, pro tvorbu grafů a dashboardů se může lišit v závislosti na verzi programu Navicat.

Jako první je třeba zvolit zdroj dat, v něm se vybere server, kde jsou data uložena, následně databáze, ve kterých jsou tabulky, se kterými je v plánu pracovat a následně i tabulky pro práci.
Data z tabulek je možné v této fázi různě upravovat pomocí různých filtrů, případně konfigurací JOINů mezi tabulkami. Jelikož jsou data pro zobrazení inflace již kompletně připravena v tabulce: „price_inflation_history“, není třeba použít více tabulek jako zdroj dat.
Pro zobrazení dat byl zvolen sloupcový graf. Sloupcový graf je velmi vizuální, což usnadňuje rychlé pochopení dat. Jde o ideální volbu pro sledování změn hodnot v čase. Každý sloupec grafu představuje jednu kategorii či jednotku a jeho výška ukazuje hodnotu, která je sledována. Sloupcový graf je velmi užitečný při sledování inflace pro různé měsíce. Výsledný graf vypadá takto:



Obrázek 1: Výstupní graf inflace vytvořený v programu Navicat
Zdroj: vlastní tvorba

V uvedeném sloupcovém grafu je vidět inflace měny v procentech za uvedené měsíce. Lze z něj vyčíst, kdy je inflace největší a zároveň, kdy jsou hráči nejaktivnější, což samo o sobě inflaci měny zhoršuje, neboť dochází k většímu vydělávání hráčů na hře a tím pádem zvětšování množství herní měny, jež je generována do systému. Pro nejlepší výsledky pro srovnávání by bylo vhodné mít data i z předchozích let. Ty však bohužel nejsou k dispozici.

Mezi nejméně aktivní měsíce patří červen, kdy se počasí oteplí a hráči nemají příliš času na hru, což se s příchodem letních prázdnin v červenci a srpnu částečně mění. Zároveň však v září je aktivita opět nižší, což má s největší pravděpodobností za následek počátek školního roku. V prosinci pak je inflace lehce záporná, vzhledem k Vánočním svátkům je aktivita hráčů malá, a také díky eventům tento měsíc jsou hráči

nuceni utrácet větší množství herní měny u NPC postav, které stahují herní měnu ze hry.

V listopadu je vidět nárust inflace měny až na hodnotu blížící se 35 %. Zde to bude pravděpodobně způsobeno výrazně větší aktivitou hráčů v souvislosti s příchodem zimy. V listopadu také probíhají každoročně dva velmi oblíbené eventy, při kterých se zároveň generuje další herní měna do systému hry. Na základě dat z grafu je evidentní, že tyto eventy budou třeba upravit, aby negenerovaly takové množství herní měny a tím nezhoršovaly ekonomickou situaci v online hře Ekura.

Pro lepší přehled a čtení v grafu je lepší porovnávat stejně měsíce v rámci roku. Za únor v roce 2022 dosahuje inflace měny 15 %. Únor byl vždy komplikovaný vzhledem k eventu, který se spouštěl každý rok na svátek Valentýn. Díky aktivitám spojeným s tímto eventem, který generoval do hry větší množství herní měny a který byl od roku 2023 zrušen.

Na základě tohoto grafu lze lépe plánovat přidávání nového herního obsahu. Herní obsah velmi často ovlivňuje ekonomiku online hry, neboť přináší hráčům nové možnosti vydělávání herní měny, zároveň však také nové úkoly, které mohou tuto měnu z oběhu hry stahovat.

Další výhodou tohoto grafu je jeho pomoc se sledováním aktivity hráčů. Jelikož je téměř nemožné určit přesný počet hráčů, kteří hrají online hru Ekura, vzhledem ke sdílení účtů, většímu počtu počítačů v domácnosti a zároveň více uživatelů zařízení. Může tento graf sloužit jako užitečná pomůcka pro plánování nového herního obsahu do hry, který není nikterak vázán na svátky. Jedná se o nový herní obsah, který má větší zásah do funkcionality hry, a taky je převážně trvalý. Herní obsahy, které jsou přidávány do hry jsou většinou na omezenou dobu, která je vázána na svátky jako Vánoce či Velikonoce.

5.6 Zhodnocení aplikovaného řešení

Na základě zpracování dat z databáze společnosti Sindicate Entertainment s.r.o., která vyvíjí online hru Ekura, je možné efektivněji řídit inflaci měny v této hře. Od zavedení tohoto nástroje a aplikace úprav cen předmětů, které jsou využívány

v novém herním obsahu dochází ke stahování herní měny z oběhu mezi hráči. Neboť herní obsah na základě dat z vytvořených tabulek získává průměrnou cenu předmětů, které následně po hráčích vyžaduje jako „oběť“ pro postup v dalším rozvoji ve hře. Někteří hráči místo vydělávání čistě herní měny v této hře se snaží tyto předměty spíše získat, přičemž již nevydělávají takové množství herní měny čímž negenerují tolik peněz. Zbylí hráči předměty kupují stále, avšak již za vyšší ceny, tudíž dochází ke stahování herní měny od hráčů, kteří jí mají nadbytek, a tak zároveň neutrácí takové množství za jiné předměty, které nejsou vhodné ke sledování, vzhledem k tomu, že jsou velmi často příliš unikátní. Jelikož utrácí za předměty, které jsou vyžadovány pro odevzdávání pro další postup, vyšší částky, následně nemají tolik herní měny, aby mohli přeplácet za jiné předměty, které si poté mohou dovolit i hráči noví.

Vzhledem k upravování cen díky tomuto nástroji dochází k motivaci hráčů, aby si některé předměty pořídili přímo od serveru za dukáty, které mohou získat pouze za reálné peníze příspěvkem společnosti, což má za následek zvýšení zisků společnosti.

V průběhu sledování dat spojenými s prodejem předmětů nebyly zaznamenány žádné extrémní výkyvy. Tudíž se dá říci, že limitace, která se zakládání na ignorování transakcí, jež jsou uskutečněny z jedné a téže IP adresy je dostatečný pro přesné hodnocení inflace cen herních předmětů. Tento postup umožňuje vyloučit potenciální zkreslení dat, které by mohlo vzniknout v důsledku přesunu herní měny mezi vlastními účty hráčů, kteří by k tomu mohli využít velmi levný předmět a za něj zaplatit daleko větší částku, než je tržní cena.

V případě, že by se v budoucnu objevily anomálie v podobě extrémních hodnot, které by se odchýlily od očekávaných trendů, bude třeba tyto data zkontrolovat a provést analýzu těchto výsledků, neboť případné anomálie by mohly signalizovat případné problémy jako je zneužívání systému hráči, kteří by mohli mít tendenci manipulovat s tržními cenami. V takovém případě by bylo nezbytné zvážit rozšíření současného systému omezení, aby zahrnovalo další identifikační prvky, jako je například PCID. Tyto dodatečné kontrolní mechanismy by mohly poskytnout další úroveň zabezpečení a pomoct předcházet případnému zkreslení dat.

Přesnější vhled na funkčnost tohoto nástroje pro sledování ekonomiky v online hře Ekura bude více pozorovatelný až po delší době používání. Nynější výsledky jsou však zatím velmi dobré a zdá se, že nástroj plní svůj účel.

Závěr

Tato bakalářská práce měla za cíl použít SQL jako nástroj pro sledování ekonomiky online hry Ekura, kterou provozuje společnost Sindicate Entertainment s.r.o. V první části této bakalářské práce se vymezily základní pojmy z oblasti počítačových her, databází, jazyka SQL, MySQL a PHP. Tyto teoretické znalostí byly následně použity při zpracování bakalářské práce.

Při práci s databází online hry Ekura, která je postavená na technologii MySQL bylo třeba identifikovat dostupná data. To zahrnovalo analýzu struktury databáze, pochopení formátů dat, které databáze obsahuje a určení, jak tyto data mohou být využita ke sledování a analýze ekonomických parametrů hry.

Následně bylo vytvořeno množství SQL dotazů, procedur a PHP skript, které umožnily extrakci a zpracování relevantních dat pro sledování inflace vybraných předmětů a následně celkové měsíční inflace měny v rámci hry. Automatizace skriptu byla provedena za pomoci tzv. Cronu.

Jedním z omezení, na které se narazilo, bylo smazání dat před rokem 2022 ze strany společnosti, která uchovávala data pro své potřeby pouze za aktuální a předchozí rok. Na základě této bakalářské práce bylo doporučeno a domluveno, že data již nebudou mazána a od počátku roku 2022 již budou kompletní. Přesto se podařilo vytvořit nástroj, na jehož základě je možné lépe vyvažovat ekonomiku online hry.

Tento nástroj může sloužit i pro plánování vydávání nového herního obsahu do online hry, kdy za jeho pomoci je možné určit, v jaké době je největší aktivita hráčů v této hře. Zároveň upozorňuje, které měsíce jsou z hlediska aktivity hráčů špatné a ve srovnání s měsíci předchozími pomáhá se srovnáním vlivu přidání nového herního obsahu v podobě eventů vázaných na svátky, a tedy každoročně opakujících se ve stejném období, na inflaci v této hře. Z čehož je možné se následně poučit a herní obsah na další rok přepracovat v případě negativního vlivu na ekonomiku či ponechat ve stavu, jakým byl.

Jedním z dalších možností, jak by šlo využít SQL jako nástroje pro sledování ekonomiky v online hře Ekura by mohlo být vytvoření nástroje, který by pomáhal

identifikovat podezřelé transakce mezi hráči. Online hra Ekura se potýká s častým porušováním podmínek použití, jelikož hráči, aby ušetřili, tak si kupují herní měnu za reálnou měnu přímo od jiných hráčů, čímž společnost přichází o peníze. K tomuto dochází právě v souvislosti s tím, že hráči hromadí herní měnu a když jí mají nadbytek, rozhodnou se inzerovat její prodej. Na toto jsou většinou nalákáni noví hráči, kteří jí mají pro změnu velmi málo.

Tento problém pomáhá částečně řešit již vytvořený nástroj v rámci této bakalářské práce, jehož výsledky jsou součástí nového herního obsahu hry a tím pádem dochází ke snižování rozdílů mezi těmito skupinami hráčů. Hráči nejsou tolik motivováni riskovat potrestání, protože nepotřebují získat takové množství měny, jelikož sledované předměty, které jsou vyžadovány v novém herním obsahu jsou snadněji získatelné v počátcích vývoje herních postav, slouží k rychlejšímu vydělávání herní měny na úkor hráčů v pozdější fázi hry. Tímto jsou motivováni si herní měnu nakoupit skrze portál společnosti, kde je to sice o něco dražší, nehrozí však riziko potrestání ze strany společnosti za porušení podmínek použití.

Výsledkem této práce je tedy nástroj, který sleduje změnu cen předmětů online hry Ekura a následně celkovou inflaci za každých 28 dní. Na základě získaných dat v průběhu zpracování je tak možné nyní plánovat či upravovat nový herní obsah.

Tato práce vyžadovala mnoho hodin práce a získávání nových znalostí. Teoretické znalosti zpracované v této bakalářské práci byly využívány téměř v každém kroku. Tento nástroj je již nyní využíván a plánuje se nasazování nových herních obsahů do hry, včetně úprav těch, které se každým rokem doposud upravovaly téměř beze změn.

Přestože byl nástroj vytvořen speciálně pro hru Ekura, jeho koncept a architektura mohou být upraveny a aplikovány pro sledování ekonomiky jiných online her. Takové uplatnění by mohlo představovat cenný přínos pro vývojáře online her.

Seznam použité literatury

BASLER, Jaromír. *Počítačové hry, jejich dělení, současné tendence vývoje a základní výzkumná šetření z oblasti počítačových her* [online]. Olomouc, 2016 [cit. 2023-02-08]. Dostupné z: https://www.researchgate.net/profile/Jaromir_Balser/publication/306039854_Pocitacove_hry_jejich_deleni_soucasne_tendence_vyvoje_a_zakladni_vyzkumna_setreni_z_oblasti_pocitacovych_her/links/57ac63bb08ae3765c3ba9fc2.pdf.

HARRINGTON, Jan L., 2016. *Relational Database Design and Implementation.* 4. vyd. Cambridge: Morgan Kaufmann. ISBN 978-0-12-804399-8.

Ian. *What does ACID mean in Databe Systems?* [online]. Database.Guide, 2016 [cit. 2023-02-17]. Dostupné z: https://database.guide/what-is-acid-in-databases/.

LAURENČÍK, Marek, 2018. *SQL: podrobný průvodce uživatele*. Praha: Grada Publishing. ISBN 978-80-271-0774-2.

LIOY, Kevin, 2019. MySQL: *SQL Darabase Programming for Beginners*. Independently piublished. ISBN 978-1688905818.

POKORNÝ, Jaroslav a Michal VALENTA, 2020. *Databázové systémy*. 2. úřeúracpvam= vydání. Praha: Česká technika – nakladatelství ČVUT. Vysokoškolská učebnice. ISBN 978-80-01-06696-6.

PROQUEST. 2021. *Databáze článků ProQuest* [online]. Ann. Arbor, MI, USA: ProQuest. [cit. 2021-09-26]. Dostupné z: http://knihovna.tul.cz/

ROTH, Jason. *SQL Sever Index Architecture and Design Guide – SQL server* [online]. 2019 [cit. 2023-03-05]. Dostupné z: https://docs.microsoft.com/en-us/sql/relational-databases/sql-server-index-design-guide?view=sql-server-2017.

STAIN, Steve. *Primary and foreign key Constraints – SQL serve* [online]. 2017 [cit. 2023-02-19]. Dostupné z: https://docs.microsoft.com/en-us/sql/relational-databases/tables/create-foreign-key-relationships?view=sql-server-2017.

TIŠŇOVSKÝ, Pavel. *Historie vývoje počítačových her: 1. část – první milníky*. In: *Root* [online]. 2011 [cit. 2023-02-08]. Dostupné z:

https://www.root.cz/clanky/historie-vyvoje-pocitacovych-her-1-cast-prvni-milniky/

VYSTAVĚL, Radek, 2021. *Databáze a SQL pro začátečníky*. Ondřejov: Radek Vystavěl. ISBN 978-80-908144-0-0.

WELLING, Luke a Laura THOMSON, 2017. *Mistrovství PHP a MySQL*. Přeložil Ondřej BAŠE. Brno: Computer Press. ISBN 978-80-251-4892-1.