Міністерство освіти і науки України

Львівський національний університет імені Івана Франка

Факультет управління фінансами та бізнесу

Кафедра цифрової економіки та бізнес-аналітики

Курсова робота

на тему :

“Інформаційна система компанії з розробки програмного забезпечення”

Студента 3 курсу, групи УФЕ-31с
напряму підготовки: Інформаційні технології в бізнесі
Крюкова Є. К..

Керівник :
к.е.н., доц. Ярема О. Р.

Національна шкала _____
Кількість балів :_____Оцінка : ECTS ____

Львів-2020

ЗМІСТ

ВСТУП.....3
РОЗДІЛ 1. ІНФОРМАЦІЙНА СИСТЕМА КОМПАНІЇ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....7
1.1 Особливості компанії.....7
1.2 Рольова модель.....8
1.3 Схема бізнес-процесу.....11
1.4 Участь ролей в операціях бізнес-процесу.....15
1.5 Інформаційна модель бізнес-процесу.....16
1.6 Засоби автоматизації бізнес-процесу.....20
РОЗДІЛ 2. ПОБУДОВА НАВЧАЛЬНОЇ БАЗИ ДАНИХ.....21
2.1 Опис предметної області.....21
2.2 Архітектура СКБД.....22
2.3 Склад таблиць бази даних.....23
2.4 Перелік таблиць баз даних.....24
2.5 Перелік полів таблиць баз даних.....24
2.6 Запити до таблиць бази даних.....26
2.7 Етап концептуального проектування.....26
ВИСНОВОК.....28
СПИСОК ВИКОРИСТАННИХ ДЖЕРЕЛ.....30

ВСТУП

Історично системи управління базами даних орієнтувалися на вирішення завдань, пов'язаних у першу чергу з транзакційною обробкою структурованої інформації. Безумовно, найкращим, перевіреним часом рішенням тут була і залишається реляційна модель СУБД. Однак в останні роки область застосування баз даних незмінно розширювалася. З одного боку, потрібно керувати більш широким набором форматів даних, переходячи до вирішення спільних проблем управління корпоративною інформацією. З іншого - саме СУБД беруть на себе основні функції інтеграції даних і додатків корпоративних систем. (За даними Gartner Group, інформаційні відділи підприємств витрачають до 40% свого бюджету на вирішення завдань інтеграції діючих компонентів баз даних.) Саме цим пояснюється активний інтерес до обговорення архітектурних принципів і можливостей реалізації баз даних різних моделей - постреляційних, об'єктно-реляційних, XML.

Останнім часом утворилися нові важливі області застосування баз даних, і кожна з них представляє принципово нове середовище, до якого необхідно адаптувати технології СУБД. Ці області отримали на ринку назви інтелектуально аналізу даних (data mining), сховищ даних (data warehousing), репозиторіїв даних (data repository).

База даних (БД) - упорядкований набір логічно взаємопов'язаних даних, що використовується спільно, та призначений для задоволення інформаційних потреб користувачів. У технічному розумінні включно й система управління БД.

Система управління базами даних (СУБД) - це комплекс програмних і мовних засобів, необхідних для створення баз даних, підтримання їх в актуальному стані та організації пошуку в них необхідної інформації.

Централізований характер управління даними в базі даних передбачає

необхідність існування деякої особи (групи осіб), на яку покладаються функції адміністрування даними, що зберігаються в базі.

Головним завданням БД є гарантоване збереження значних обсягів інформації та надання доступу до неї користувачеві або ж прикладній програмі. Таким чином БД складається з двох частин: збереженої інформації та системи управління нею. З метою забезпечення ефективності доступу записи даних організовують як множину фактів (елемент даних).

Існує величезна кількість різновидів баз даних, що відрізняються за критеріями (наприклад, в Енциклопедії технологій баз даних визначаються понад 50 видів БД).

Класифікація БД *за* моделлю даних:

- ієрархічні,
- мережеві,
- реляційні,
- об'єктні,
- об'єктно-орієнтовані,
- об'єктно-реляційні.

Окреме місце в теорії та практиці займають просторові, тимчасові, або темпоральні (temporal) і просторово-часові (spatial-temporal) БД.

Ієрархічні бази даних можуть бути представлені як дерево, що складається з об'єктів різних рівнів. Верхній рівень займає один об'єкт, другий - об'єкти другого рівня і т.д.

Між об'єктами існують зв'язки, кожен об'єкт може включати в себе декілька об'єктів більш низького рівня. Такі об'єкти перебувають у відношенні предка (об'єкт більш близький до кореня) до нащадка (об'єкт більш низького рівня), при

цьому можлива ситуація, коли об'єкт-предок не має нащадків або має їх декілька, тоді як у об'єкта-нащадка обов'язково тільки один предок. Об'єкти, що мають загального предка, називаються близнюками.

Мережеві бази даних подібні до ієрархічних, за винятком того, що в них є покажчики в обох напрямках, які з'єднують споріднену інформацію.

До основних понять мережевої моделі бази даних відносяться: рівень, елемент (вузол), зв'язок.

Вузол - це сукупність атрибутів даних, що описують деякий об'єкт. На схемі ієрархічного дерева вузли представляються вершинами графа. У мережній структурі кожен елемент може бути пов'язаний з будь-яким іншим елементом.

Незважаючи на те, що ця модель вирішує деякі проблеми, пов'язані з ієрархічною моделлю, виконання простих запитів залишається досить складним процесом.

Також, оскільки логіка процедури вибірки даних залежить від фізичної організації цих даних, то ця модель не є повністю незалежною від програми. Іншими словами, якщо необхідно змінити структуру даних, то потрібно змінити і додаток.

Реляційна модель орієнтована на організацію даних у вигляді двовимірних таблиць. Кожна реляційна таблиця являє собою двовимірний масив.

Об'єктна СУБД ідеально підходить для інтерпретації складних даних, на відміну від реляційних СУБД, де додавання нового типу даних досягається ціною втрати продуктивності або за рахунок різкого збільшення термінів і вартості розробки додатків. Об'єктна база, на відміну від реляційної, не вимагає модифікації ядра при додаванні нового типу даних. Новий клас і його екземпляри просто надходять у зовнішні структури бази даних. Система управління ними залишається без змін.

Об'єктно-орієнтована база даних (ООБД) - база даних, в якій дані оформлені у вигляді моделей об'єктів, що включають прикладні програми, які управляються зовнішніми подіями. Результатом поєднання можливостей (особливостей) баз

даних і можливостей об'єктно-орієнтованих мов програмування є об'єктно-орієнтовані системи управління базами даних (ООСУБД). ООСУБД дозволяють працювати з об'єктами баз даних також, як з об'єктами у програмуванні в об'єктно-орієнтованих мовах програмування. ООСУБД розширює мови програмування, прозоро вводячи довготривалі дані, управління паралелізмом, відновлення даних, асоційовані запити й інші можливості.

Об'єктно-реляційні системи поєднують переваги сучасних об'єктно-орієнтованих мов програмування з такими властивостями реляційних систем як множинні представлення даних і високорівневі непроцедурні мови запитів.

РОЗДІЛ 1. ІНФОРМАЦІЙНА СИСТЕМА КОМПАНІЇ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Особливості компанії

В індустрії розробки програмного забезпечення (ПЗ) існує багато різних методологій розробки, які вдають із себе досить концептуальні бачення того, як слід реалізовувати проекти по створенню ПЗ. Як приклади таких методологій можна привести: Rational Unified Process (RUP), Microsoft Solution Framework (MSF), Extreme Programming (XP), Agile, Capability Maturity Model Integration (CMMI) і багато інших.

Поділяючи важливість методології як основи для реальних бізнес-процесів, слід зазначити різницю в поняттях методологія і бізнес-процес. Бізнес-процес являє собою реалізацію методології, її окремих елементів або елементів декількох методологій в конкретній організації при виконанні конкретних проектів і створенні конкретних продуктів. Тому, створення бізнес-процесу розробки ПЗ на основі однієї з існуючих методологій є першою і найважливішою задачею компанії, яка бажає зайнятися в тому чи іншому вигляді створенням софта (для зовнішніх замовників або своїх внутрішніх потреб).

Як показує досвід, цілком можливо створити невелику групу розробки і почати робити софт і без чіткого опису бізнес-процесу. Однак, коли число учасників такої «неформальної» команди стає більше п'яти чоловік, то втрати від відсутності чіткого регламенту стають незрівнянно більшими в порівнянні з витратами на регламентацію бізнес-процесу і спеціалізоване ПО для його автоматизації. Втрати в даному випадку можуть бути як прямими (наприклад, нескінченні переробки однієї і тієї ж функціональності через невідповідність вимогам замовника), так і непрямими (наприклад, погіршення психологічної атмосфери в колективі,

пов'язане з нерозумінням зони своєї відповідальності кожним учасником команди).

Попробуємо навести приклад бізнес-процесу розробки ПЗ на основі елементів декількох методологій (найбільша кількість елементів взято з MSF) і багаторічного досвіду розробки та управління розробкою ПЗ інших людей. Даний бізнес-процес орієнтований на ведення великих проектів з розробки ПЗ на досить «зрілій» стадії, коли продукт вже може експлуатуватися замовниками і коли мова вже йде швидше про розвиток і доробку функціоналу, а також усунення «багів», ніж про розробку «з нуля» невеликих програмних продуктів.

1.2 Рольова модель

Найменування ролі: Керівник проекту

Зона відповідальності:

- Формування планів
- Контроль виконання планів
- Організаційна робота (в тому числі і з Замовником)
- Концептуальна архітектура рішення
- Частина аналітичної роботи

Найменування ролі: Керівник групи

Зона відповідальності:

- Оцінка тривалості і трудомісткості завдань в процесі планування
- Контроль виконання планів групою
- Розподіл робіт всередині групи
- Концептуальна архітектура рішення

- Частина аналітичної роботи
- Організація збору вимог замовника
- Відповідність діяльності групи бізнес-процесу розробки
- Робота групи з замовником

Найменування ролі: Аналітик

Зона відповідальності:

- Збір вимог замовника
- Розробка ТД на функціональність
- Розробка планів тестування
- Концептуальне тестування функціональності
- Розробка документації для користувачів

Найменування ролі: Архітектор

Зона відповідальності:

- Архітектура рішення і відповідність її вимогам до вирішення
- Розробка РП на функціональність (визначає принципові моменти, в подальшому їх деталізує в рамках РП Розробник)
- Контроль якості коду і відповідність його проектним рішенням по архітектурі
- Репозиторій інформації з архітектури рішення
- Бере участь у формуванні планів і оцінки складності та тривалості завдань
- Бере участь у комплексному тестуванні

Найменування ролі: Розробник

Зона відповідальності:

- Розробка РП (за участю Архітектора в процесі вироблення принципових рішень)
- Розробка функціональності
- Якість коду
- Виправлення помилок в коді
- Проведення первинного тестування коду
- Бере участь у комплексному тестуванні коду

Найменування ролі: Тестер

Зона відповідальності:

- Тестування функціональності
- Написання Unit тестів
- Бере участь у розробці планів тестування

Найменування ролі: Білд-інженер

Зона відповідальності:

- Збірка версії
- Випуск версії (після тестування)
- Підготовка супровідних документів до версії

Також, один фахівець може виконувати декілька ролей, але з урахуванням певних обмежень.

1.3 Схема бізнес-процесу

Основна схема бізнес-процесу приведена нижче. У ній беруть участь члени проектної команди згідно рольової моделі.

Бізнес-процес включає в себе п'ять груп (контурів) робіт:

- Контур збору вимог
- Контур середньострокового планування
- Контур аналітичних робіт
- Контур розробки версії (ітерація)
- Контур невеликих доробок

Контур збору вимог

Зокрема, вимогами є:

1) Помилки, виявлені при внутрішньому або зовнішньому (ПСІ) тестуванні, в процесі експлуатації рішення

2) Функціональні вимоги (доопрацювання)

3) Нефункціональні вимоги (обмеження, яким має задовольняти рішення)

Вимоги можуть надходити з різних джерел, наприклад:

1) Представники Замовника

2) Тестери в складі проектної команди

3) Керівництво компанії

Контур середньострокового планування

В рамках контуру збору вимог членами проектної команди шляхом проведення наради приймається рішення про реалізацію підстави вимог які є. Проводиться їх пріоритезація і вирішується, в яку саме версію повинні потрапити вимоги в залежності від їх пріоритету.

Розподіл вимог відбувається за кількома наспутними версіями з часовим інтервалом до півроку. В результаті складається середньостроковий план, який представляє з себе список вимог, які планується реалізувати в наступних версіях. У картці кожного запланованого вимоги позначається ця версія.

На підставі середньострокового плану по реалізації вимог проводиться детальне планування аналітичних робіт з написання ТД з постановкою завдання по їх реалізації.

Контур аналітичних робіт

Згідно з планом робіт Аналітик виробляє аналітичне опрацювання вимоги і складає документ з затвердженою структурою розділів - Технічний проект з описом логіки реалізації вимоги.

Технічний проект, підготовлений на посаді аналітика, узгоджується з:

Керівником проекту

Керівником групи

Архітектором

Після складання і узгодження ТД вимога позначається як готова до включення в план розробки версії.

Контур розробки версії

Контур розробки версії складає з себе одну ітерацію розробки: від планування робіт за версією, до її випуску. Після випуску однієї версії, починаються робіт по наступній версії.

При плануванні робіт за версією проектна команда переглядає вимоги, вибираючи серед них ті, які:

1) Відповідно до середньострокового плану повинні бути реалізовані в рамках цієї версії

2) За якими завершена аналітична проробка (є погоджений ТД)

Також до складу робіт за версією можуть включатися вимоги, які мають критичну важливість і найвищий пріоритет. В цьому випадку їх аналітичне опрацювання планується в рамках робіт за версією. Однак цей варіант не є основним.

Після виділення вимог, що підлягають розробці в рамках справжньої версії, складається детальний план розробки цієї версії, що включає в себе всі види робіт.

Потім згідно з планом Розробник спільно з Архітектором (в частині концептуальних архітектурних рішень) на підставі Технічного проекту пише Робочий проект, в якому описує реалізацію відповідної вимоги.

Робочий проект, підготовлений Розробником, узгоджується з:

Архітектором

Керівником групи

Керівником проекту

Контур невеликих доробок

Невеликі доопрацювання - доопрацювання з тривалістю реалізації менше людино-дня, які не потребують написання Технічного проекту і робочого проекту.

Всі невеликі доопрацювання діляться на дві групи:

З нормальним пріоритетом - включаються в наступну основну версію.

З критичним пріоритетом - для них випускається проміжна версія шляхом внесення необхідних змін до екземпляру з кодом поточної розгорнутої версії.

Рішення про реалізацію дрібного доопрацювання з нормальним пріоритетом приймається Керівником групи шляхом аналізу завантаження розробників по групі. Найменш завантажені на реалізації планових доробок за версією Розробники займаються усуненням дрібних доопрацювань з нормальним пріоритетом.

Після усунення дрібних доопрацювань з нормальним пріоритетом вони потрапляють в робочу базу коду і випускаються з наступною основною версією.

Рішення про реалізацію дрібного доопрацювання з критичним пріоритетом приймається Керівником проекту спільно з Керівником групи. В цьому випадку Розробник, найбільш добре знайомий з проблемою яка виникла може бути тимчасово переключений з планових робіт на її усунення.

Після усунення критичних дрібних доопрацювань (одного або декількох відразу) ініціюється процедура випуску проміжної версії. Збірку здійснює Білд-інженер, також він готує Супровідний документ в якому вказує який номер даної проміжної версії і перелік критичних дрібних доопрацювань усунених в ній.

Після випуску проміжної версії Розробники вносять зміни, відповідні критичним дрібним доопрацюванням, в основну робочу версію.

1.4 Участь ролей в операціях бізнес-процесу

Одна з основних вимог до бізнес-процесу розробки - рівномірне завантаження всіх членів проектної команди на всьому протязі розробки з урахуванням її ітеративного і послідовного (ТД, РП, реалізація і т.д.) характеру.

На схемі нижче як приклад наведено розподіл робіт при розробці версії з тривалістю 8 тижнів. Розглянута вся проектна команда, задіяна в усіх контурах робіт.

Ключовим елементом даного бізнес-процесу, проілюстрованим на схемі, є незалежне виконання аналітичних робіт (Технічний проект) і робіт по розробці версії (Робочий проект і реалізація) один від одного з метою забезпечення рівномірного завантаження Аналітика на всій ітерації розробки.

Також важливим моментом є участь Розробника, як в планових роботах по версії, так і в усуненні дрібних, в т.ч. високо критичних зауважень.

Тестер на ранніх стадіях ітерації займається тестуванням попередньої версії і реєструє виявлені помилки в Репозиторії вимог, пізніше в міру готовності версії він переключається цілком на тестування поточної версії.

Білд-інженер має невисоке завантаження на більшій частині ітерації, тому, доцільно поєднання цієї ролі з однією з наступних ролей: Розробник, Керівник групи, Архітектор.

Найважливішим завданням Архітектора крім участі в написанні Робочих проектів і контролю технічного боку проведення розробки є також ведення

Репозиторія архітектури - поновлення його в міру розробки нової функціональності відповідно до Робочими проектами.

Керівник проекту і керівник групи крім задач по управлінню проектом і групою відповідно займаються аналітичною роботою, особливо концептуальним проектуванням рішення.

1.5 Інформаційна модель бізнес-процесу

Всі види інформації в рамках бізнес-процесу розробки розподіляються по п'яти сховищ (репозиторіїв):

- Репозиторій вимог
- Репозиторій архітектури
- Репозиторій документації
- Репозиторій коду
- Репозиторій завдань (план робіт)

Елементи в одному сховищі можуть бути пов'язані з елементами в іншому сховищі. Наприклад, вимоги в репозиторії вимог можуть бути пов'язані з компонентами з репозиторія архітектури, якими вони реалізуються.

На схемі нижче показані приклади вмісту репозиторіїв і взаємозв'язку між ними. Потім в таблиці дається коротке пояснення по змісту репозиторіїв і описані взаємозв'язки між їх елементами.

Найменування репозиторія: Репозиторій вимог

Короткий опис:

Ієрархічна структура груп, що містить вимоги до вирішення, що є вихідними підставами для будь-яких робіт з розробки:

Функціональні вимоги (функції рішення)

Нефункціональні вимоги

Помилки (баг-трекінг)

Групи можуть бути вкладеними. Вимоги не можуть бути вкладеними.

Кожна вимога містить наступну інформацію:

Найменування

Короткий опис

Поточний статус (відображає поточний етап життєвого циклу вимоги)

Тип вимоги (Розширення / Помилка)

Джерело (Хто повідомив про вимогу)

Підстава (посилання на договірні документи, ТЗ і інші зобов'язання)

Пріоритет

Версія (в якій планується реалізувати вимогу)

Статуси вимог змінюються вручну відповідними ролями по завершенні етапів робіт відповідно до бізнес-процесом.

З кожним вимогою можуть бути асоційовані елементи з наступних репозиторіїв:

Репозиторій архітектури - компоненти, що реалізують цю вимогу

Репозиторій документації - документи, що описують дану вимогу (ТД, РП, тести та ін.)

Репозиторій завдань - завдання в плані робіт, шляхом виконання яких реалізується дана вимога

Найменування репозиторія: Репозиторій архітектури

Короткий опис:

Сховище з описом архітектури. Включає в себе наступні елементи:

Діаграма компонентів 1-го рівня. На ній представлені шари, на які поділено рішення і середовища, в рамках яких ці шари функціонують. Усередині шару зображуються пакети, з яких він складається.

Діаграми компонентів 2-го рівня. Для кожного пакета на діаграмі першого рівня складається одна діаграма компонентів з переліком компонентів, що входять до складу цього пакета.

Опис компонентів. Для кожного компонента на діаграмі 2-го рівня готується документ з описом його інтерфейсів і внутрішньої логіки реалізації.

Довідник класів. Автоматично генерований за кодом навігатор по класам, реалізованих в системі.

З кожним пакетом або компонентом на моделях 2-го рівня в репозиторії архітектури можуть бути асоційовані елементи з наступних репозиторіїв:

Репозиторій вимог - вимоги, при реалізації якого використовується даний компонент.

Репозиторій документації - загальні описи архітектури, опису шарів, компонентів (інтерфейсні і внутрішні частини), класів.

Репозиторій коду - посилання на конкретні програмні модулі, що реалізують даний компонент.

Найменування репозиторія: Репозиторій документації

Короткий опис:

Сховище файлів різних типів, які використовуються як самі по собі (наприклад, Договірні документи, Протоколи, Акти та ін.), так і в зв'язці з елементами зі сховищ вимог (ТД, РП) і сховища архітектури (опису компонентів, шарів).

Найменування репозиторія: Репозиторій коду

Короткий опис:

Весь код рішення з підтримкою розгалуження версій.

Для цілей термінової реалізації критичних доопрацювань в рамках Репозиторія коду існують два примірника коду рішення:

Робоча версія - в ній виробляються всі доопрацювання згідно плану робіт за версією і дрібні доопрацювання нормального пріоритету.

Поточна розгорнута версія - експлуатована зараз версія - в ній реалізуються критичні дрібні доопрацювання і випускається проміжна версія кожен раз, коли виникає необхідність усунути якісь принципові моменти не чекаючи випуску наступної версії.

Після випуску проміжної версії, вносяться відповідні зміни в Робочу версію в напівавтоматичному режимі.

Найменування репозиторія: Репозиторій завдань (план робіт)

Короткий опис:

План робіт, що містить завдання для всіх учасників проекту. Завдання пов'язані з вимогами, на реалізацію яких вони спрямовані. Також завдання пов'язані з

CheckIn`гами коду, який створювався або модифікувався в рамках виконання завдання.

1.6 Засоби автоматизації бізнес-процесу

Найменування програмного продукту: Microsoft Team Foundation Server (MS TFS)

Спосіб використання в бізнес-процесі:

1) Репозиторій вимог

2) Репозиторій завдань

3) Репозиторій коду

4) Репозиторій документів

Найменування програмного продукту: Microsoft Visual Studio 2008

Спосіб використання в бізнес-процесі:

1) Засіб розробки коду

2) Клієнт для MS TFS

Найменування програмного продукту: Microsoft Visio

Спосіб використання в бізнес-процесі:

1) Репозиторій архітектури (моделі 1-го і 2-го рівнів)

Найменування програмного продукту: Doxygen

Спосіб використання в бізнес-процесі:

1) Репозиторій архітектури (довідник класів)

Найменування програмного продукту: Microsoft Project

Спосіб використання в бізнес-процесі:

1) Клієнт для сховища завдань в TFS

Найменування програмного продукту: Microsoft Office

Спосіб використання в бізнес-процесі: 1) Засіб створення документації

РОЗДІЛ 2. ПОБУДОВА НАВЧАЛЬНОЇ БАЗИ ДАНИХ

2.1 Опис предметної області

У сучасних інформаційних системах для забезпечення роботи з базами даних використовують системи керування базами даних (СКБД). Система керування базами даних — це система, заснована на програмних та технічних засобах, яка забезпечує визначення, створення, маніпулювання, контроль, керування та використання баз даних. Застосунки для роботи з базою даних можуть бути частиною СКБД або автономними. Найпопулярнішими СКБД є MySQL, PostgreSQL, Microsoft SQL Server, Oracle, Sybase, Interbase, Firebird та IBM DB2. СКБД дозволяють ефективно працювати з базами даних, обсяг яких робить неможливим їх ручне опрацювання.

Через тісний зв'язок баз даних з СКБД під терміном «база даних» інколи необґрунтовано та неточно мають на увазі систему керування базами даних. Але варто розрізняти базу даних — сховище даних, та СКБД — засоби для роботи з базою даних. СКБД з інформаційної системи може бути видалена, але база даних продовжить існувати. І навпаки: СКБД може функціонувати без жодної бази даних.

В загальному базу даних неможливо просто перемістити з однієї СКБД до іншої. Але використовують стандарти (SQL, ODBC, JDBC), які уніфікують ряд операцій по роботі з даними і СКБД дозволяють різним застосункам працювати з базами даних різних СКБД. СКБД часто класифікують за моделлю організації даних. Найвживаніші СКБД використовують реляційну модель, у якій дані подають у виді таблиць. Для кінцевого користувача (та прикладних програм) робота з базою даних напряму неможлива. Всі маніпуляції над даними здійснюють через

спеціальні запити, які надсилають до СКБД. СКБД опрацьовує їх і повертає результат. Безпосередньо з базою даних працює виключно СКБД.

Сучасні СКБД забезпечують функції щодо керування даними, які можна поділити на такі групи:

• Оголошення даних — створення, зміна та видалення визначень які описують організацію даних.

• Модифікація даних — додавання даних, їх редагування та видалення.

• Отримання даних — надання даних за запитом застосунку у формі, яка дозволяє їх безпосереднє використання. Дані можуть надаватись або у формі, в якій вони зберігаються у базі даних, або в іншій формі (наприклад, через поєднання різних даних).

• Адміністрування даних — реєстрування та відслідковування дій користувачів, дотримання безпеки роботи з даними, забезпечення надійності та цілісності даних, моніторинг продуктивності, резервне копіювання та відновлення даних тощо.

2.2 Архітектура СКБД

Ця архітектура передбачає виділення однієї з машин мережі як головної (сервер). На такій машині зберігається спільна централізована БД. Усі інші машини мережі виконують функції робочих станцій, за допомогою яких підтримується доступ користувацької системи до бази даних. Файли бази даних відповідно до призначених для користувача запитів передаються на робочі станції, де в основному і проводиться обробка даних. При великій інтенсивності доступу до одних і тих же даних продуктивність інформаційної системи різко падає. Користувачі також можуть створювати на робочих станціях локальні БД, які використовуються ними монопольно.

У сучасних мережевих інформаційних системах для роботи із загальною базою даних використовують архітектуру «клієнт-сервер». При цьому в мережі

розміщують сервер баз даних. Ним виступає комп'ютер (або комп'ютери), який містить бази даних, СКБД та пов'язане з ними програмне забезпечення, і налаштований для надання користувачам інформаційної системи доступу до бази даних. Клієнти, які працюють із даними (вони можуть бути розташовані на різних комп'ютерах мережі), надсилають відповідні запити серверу. Сервер їх отримує, опрацьовує, та надсилає відповідь клієнту. Сучасні СКБД (MySQL, PostgreSQL, Microsoft SQL Server та інші) працюють відповідно до цієї архітектури. Сервер баз даних, як правило, є достатньо потужною багатопроцесорною системою, яка використовує масиви дисків RAID для підвищення надійності зберігання даних. Використання дискових масивів RAID дозволяє відновити дані, навіть якщо один або декілька з дисків вийшли з ладу.

2.3 Склад таблиць бази даних

VARCHAR - тип змінної довжини. У такій колонці рядок буде займати рівно свою довжину (в кількості символів). Однак MySQL додасть ще 1 або 2 байти на зберігання довжини самого рядка. Також варто врахувати, що оновлення такого рядка може бути дорогою операцією (загрожує фрагментацією даних, а значить - уповільненням читання).

У той час, як поле типу CHAR завжди може розподілити пам'ять для максимального числа символів, яке може зберігатися в полі, поле VARCHAR при будь-якій кількості символів може розподілити тільки певну кількість пам'яті, щоб зберегти фактичний зміст поля, хоча SQL може встановити деякий додатковий простір пам'яті, щоб стежити за поточною довжиною поля. Поле VARCHAR може бути будь-якої довжини, включаючи реалізаційну - яка визначає максимум. Цей максимум може змінюватися від 254 до 2048 символів для VARCHAR і до 16000 символів для LONG. LONG зазвичай використовується для тексту пояснювального характеру або для даних, які не можуть легко стискуватися в прості значення

полів; VARCHAR може використовуватися для будь-якого текстового рядка, чия довжина може змінюватися.

Витяг і модифікування полів VARCHAR - більш складний, і, отже, більш повільний процес, ніж вилучення та зміну полів CHAR. Крім того, деяка кількість пам'яті VARCHAR, залишається завжди невикористаною для гарантії вміщення всієї довжини рядка. При використанні таких типів слід передбачати можливість полів до об'єднання з іншими полями.

Тип даних DATE приймає значення дати. При оголошенні типу даних DATE параметри не потрібні. Значення дати повинні бути вказані в формі: РРРР-ММ-ДД. Однак PointBase також буде приймати однозначні записи для значень місяця і дня. Значення місяця повинні бути від 1 до 12, значення дня - від 1 до 31, в залежності від місяця, а значення року - від 0 до 9999.

Значення, присвоєні типу даних DATE, повинні бути укладені в одинарні лапки, перед якими стоїть ключове слово DATE без урахування регістру; наприклад, ДАТА '1999-04-04'. INT(size) Діапазон від -2 147 483 648 до 2 147 483 647

2.4 Перелік таблиць баз даних

Таблицями бази даних є:

1. Software_company – таблиця, в якій містяться загальні дані про компанію
2. Office – таблиця, в якій містяться дані про офіс компанії
3. Services – таблиця, в якій містяться дані про послуги які надає компанія
4. Ordering – таблиця, в якій містяться дані про замовлення
5. Bill – таблиця, в якій містяться дані про чеки

2.5 Перелік полів таблиць баз даних

Поля, які ідентифікують властивості таблиці «Software_company»:

1. software_company_id – айді компанії
2. website – веб-сайт компанії
3. name_of_company – назва компанії
4. address – адреса компанії

Поля, які ідентифікують властивості таблиці «Office»:

1. office_id – айді офісу
2. address – адрес офісу
3. counter_of_workers – список людей які працюють

Поля, які ідентифікують властивості таблиці «Services»:

1. services_id – айді послуги
2. name – назва послуги
3. price – ціна послуги
4. deadline – час виконання послуги
5. date – дата початку роботи послуги

Поля, які ідентифікують властивості таблиці «Ordering»:

1. ordering_id – айді замовлення
2. services_id – айді послуги замовлення
3. deadline – час виконання послуги замовлення
4. price – ціна послуги замовлення
5. date – час початку роботи послуги замовлення

Поля, які ідентифікують властивості таблиці «Bill»**:**

1. bill_id – айді чеку

2. ordering_id – айді замовлення в чеку
3. price – ціна послуги замовлення в чеку
4. deadline – час виконання роботи послуги замовлення в чеку
5. date – час початку роботи послуги замовлення в чеку

2.6 Запити до таблиць бази даних

DROP TABLE - видалення таблиці

SELECT * FROM -вивід таблиці

CREATE TABLE – створення таблиці

INT AUTO_INCREMENT PRIMARY KEY,

price FLOAT,

name VARCHAR(30)

ordering_id INT

adress VARCHAR(30)

INSERT INTO bill VALUES - наповнення полів таблиці

USE – продовжити працювати в даній базі дані

2.7 Етап концептуального проектування

Етап концептуального проектування полягає в описі і синтезі інформаційних вимог користувачів у початковий проект БД. Вихідними даними можуть бути сукупність документів користувача при класичному підході або алгоритми додатків (алгоритми бізнесу) при сучасному підході. Результатом цього етапу є високорівневе подання (у вигляді системи таблиць БД) інформаційних вимог користувачів на основі різних підходів.

Спочатку вибирається модель БД. Потім створюється структура БД, яка заповнюється даними за допомогою систем меню, екранних форм або в режимі перегляду таблиць БД. Тут же забезпечується захист і цілісність (у тому числі посилальна) даних за допомогою СУБД або шляхом побудови тригерів.

Концептуальне проектування бази даних - процес створення моделі використовуваної на підприємстві інформації, що не залежить від будь-яких фізичних аспектів її представлення. Перша фаза процесу проектування бази даних називається концептуальним проектуванням бази даних. Вона полягає в створенні концептуальної моделі даних для аналізованої частини підприємства. Ця модель даних створюється на основі інформації, записаної в специфікаціях вимог користувачів. Концептуальне проектування бази даних абсолютно не залежить від таких подробиць її реалізації, як тип обраної цільовий СКБД, набір створюваних прикладних програм, використовувані мови програмування, тип обраної обчислювальної платформи, а також від будь-яких інших особливостей фізичної реалізації. При розробці концептуальна модель даних постійно піддається тестуванню і перевірці на відповідність вимогам користувачів. Створена концептуальна модель дані підприємства є джерелом інформації для фази логічного проектування бази даних. Приступаючи до розроблення локальної концептуальної моделі даних для представлення користувача «Керівник проекту» та «Головний інженер» у базі даних «Компанія з розробки програмного забезпечення», насамперед, варто виявити різні компоненти цієї моделі, використовуючи наявні специфікації вимог користувача.

У кожну створювану модель даних входять наступні компоненти:

Типи сутностей;

Типи зв'язків;

Атрибути;

Домени атрибутів;

Потенційні ключі.

ВИСНОВОК

Рано чи пізно, в залежності від темпів розвитку, система управління будь-якої організації ускладнюється настільки, що стає складно обійтися без якісного програмного забезпечення, що дозволяє спростити і навіть автоматизувати бізнес-процеси. Для того щоб ефективно контролювати відбуваються на підприємстві події, своєчасно інформувати співробітників про всі зміни в стратегії компанії створено спеціальне програмне забезпечення для бізнесу.

Щоб програмне забезпечення бізнес-планування найбільш повно задовольняло вимоги сучасних організацій, розробники ПЗ повинні виконувати певні бізнес-вимоги до програмного забезпечення.

Програмне забезпечення аналізу та розробки бізнес-планів дає можливість спрогнозувати успіх бізнес-проектів, розрахувати терміни виконання поставлених завдань і їх прямий вплив на діяльність компанії. Щоб грамотно скласти бізнес-план, програмне забезпечення повинно мати достатній функціонал для повноцінного аналізу та прогнозування діяльності організації.

Програмне забезпечення бізнес-процесів - невід'ємна частина діяльності підприємства, яка спрямована на автоматизацію безлічі завдань, підвищення ефективності роботи організації і моніторинг актуальних проблем, які потребують термінового вирішення.

Програмні продукти компанії 1С на сьогоднішній день необхідні фактично кожній організації, яка прагне дійти до успішної діяльності та стабільного розвитку. Програма 1С: Управління торгівлею призначена для компаній будь-

якого розміру і виду діяльності. Завдяки універсальному характеру ПЗ, її можна застосовувати для аналізу роботи і автоматизації обліку на підприємствах, що займаються різними видами торгівлі.

Microsoft Project - функціональне рішення, що дозволяє здійснювати менеджмент проектів і оптимізувати ресурси, виділені на здійснення завдань. Програма добре справляється з моніторингом виконання проектів і аналізує обсяги майбутньої роботи. ПЗ можна адаптувати відповідно до потреб організації, вибравши потрібну конфігурацію.

Серед програмного забезпечення для автоматизації бізнес процесів Microsoft Dynamics CRM виділяється якісними інструментами для ефективної організації бізнес-процесів і їх автоматизації. Кілька тематичних конфігурацій програми дають можливість раціонально розподілити ресурси, автоматизувати процеси і спростити процес створення і обробки замовлень, ринкових пропозицій і багато іншого.

Pilot-ICE - надійне рішення для організацій, що займаються проектною діяльністю, яке дозволяє автоматизувати бізнес-процеси за кількома напрямками і здійснювати менеджмент корпоративних даних. Функціональний інструмент 1С-Бітрікс: Корпоративний портал пропонує безліч варіантів управління діяльністю компанії, в тому числі автоматизацію бізнес-процесів, контроль над персоналом і грамотний розподіл ресурсів.

Серед продуктивного ПЗ для оптимізації бізнесу особливо виділяється потужна платформа Red Hat JBoss Enterprise BRMS Platform. Продукт включає в себе надзвичайно широкий спектр можливостей, включаючи можливість моделювання бізнес-правил, контроль над автоматизацією бізнес-процесів, починаючи з їх створення і закінчуючи контролем за їх реалізацією, а також безліч функцій для розробників додатків.

СПИСОК ВИКОРИСТАННИХ ДЖЕРЕЛ

Алан Бьюли "Изучаем SQL" (2007)

Крис Фиайли "SQL" (2013)

Энтони Молинаро "SQL. Сборник рецептов" (2009)

Алекс Кригель и др. "SQL. Библия пользователя", 2-е издание (2010)

Эрик Редмонд, Джим Р. Уилсон "Семь баз данных за семь недель. Введение в современные базы данных и идеологию NoSQL" (2015)

А. Ю. Васильев "Работа с PostgreSQL: настройка и масштабирование"