МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

"ЗАТВЕРДЖУЮ"
Заступник керівника
(проректор з науково-педагогічної роботи)

М. В. Афанасьєв

Технології баз даних:

робоча програма навчальної дисципліни

Галузь знань	**12 «інформаційні технології»**
Спеціальність	**усі**
Освітній рівень	**перший (бакалаврський)**
Освітня програма	**усі**

Вид дисципліни	**вибіркова**
Мова викладання, навчання та оцінювання	**українська**

Завідувач кафедри інформаційних систем Руденко О. Г.

Харків
ХНЕУ ім. С. Кузнеця
2018

ЗАТВЕРДЖЕНО
на засіданні кафедри інформаційних систем
Протокол № 1 від 27.08.2018 р.

Розробники:
Федько В. В., к. ф.-м. н., доц. кафедри інформаційних систем,
Лосєв М. Ю., к. т. н., доц. кафедри інформаційних систем

Лист оновлення та перезатвердження робочої програми навчальної дисципліни

Навчальний рік	Дата засідання кафедри – розробника РПНД	Номер протоколу	Підпис завідувача кафедри

1. Вступ

Анотація навчальної дисципліни:

Сучасні економічні умови господарювання вимагають від фахівців, незалежно від їх спеціалізації, всебічного використання новітніх інформаційних технологій, комп'ютеризованих засобів збору, обробки та надання необхідної інформації. Метою цих технологій є значне підвищення якості та оперативності економічних розрахунків, намагання зробити значно ефективнішим процес обґрунтування економічних рішень тощо. Крім того, широке розповсюдження інформаційних технологій висунуло на передній план задачу створення зручного інтерфейсу користувача. Передові комп’ютерні компанії почали боротьбу за кінцевого користувача їх продукції. У цьому контексті навчальна дисципліна "Технології баз даних" є однією з найважливіших. Вона відноситься до дисциплін за вибором і становить той фундамент, на якому базується проектування та безпосередньо створення програмних продуктів у бізнесі.

Характерною рисою переважної більшості програмних продуктів є використання даних, що зберігаються у базах даних. Тому технології доступу до даних стали важливою частиною розробки застосувань і є невід’ємним напрямком підготовки сучасних фахівців у галузі комп’ютерних наук. Ураховуючи те, що ефективність праці розробника значною мірою залежить від технологічності засобів, які використовуються під час створення застосувань, проблема опанування сучасних засобів доступу до даних набула актуальності протягом останніх років. Тому оволодіння сучасними та перспективними технологіями є важливою складовою у підготовці фахівців із розробки програмного забезпечення.

Дана навчальна дисципліна призначена для студентів галузі знань 12 "Інформаційні технології" для формування у студентів компетентностей, які забезпечуються

отриманням концептуальних сучасних знань і вмінь розв’язувати складні непередбачувані задачі, створенням у тих, хто навчається, спроможності донесення до фахівців власного досвіду, а також відповідальності за прийняття рішень у різноманітних умовах.

Навчальна дисципліна "Технологія баз даних" вивчається згідно з навчальним планом підготовки фахівців освітнього рівня "бакалавр" галузі знань 12 "Інформаційні технології" для денної форми навчання.

Робоча програма навчальної дисципліни «Технології баз даних» складається з таких змістових модулів:

1. Класичні засоби доступу до даних.
2. Сучасні засоби доступу до даних.

Знання, що отримані під час вивчення першого змістового модуля в теоретичному плані потрібні для розуміння підходів використання баз даних в застосування, які створюються різними мовами програмування (C#, Java, PHP, Python тощо). Практичні вміння та навички дозволять розробнику супроводжувати застосування, що створювалися п’ять і більше років тому назад. Він охоплює такі теми:

з'єднання з базами даних в ADO.NET. Виконання операцій у з'єднаному середовищі;

виконання операцій у роз'єднаному середовищі;

JDBC – доступ до даних у Java.

Знання, вміння та навички, що отримані під час вивчення другого змістового модуля потрібні для створення сучасних застосувань (настільних та веб) в IT компаніях, в яких працюватимуть спеціалісти після завершення навчання в університеті. Цей модуль охоплює такі теми:

типізовані набори даних;

технологія LINQ to DataSet;

платформа Entity Framework. Технологія Code First;

побудова звітів (Reporting);

перспективи розвитку баз даних та технологій доступу до них.

Мета навчальної дисципліни:

Метою викладання даної навчальної є формування у студентів навичок розробки та супроводження програм, що використовують дані, які зберігаються в базах даних, програмна реалізація CRUD-операцій з базами даних, аналіз даних та побудова звітів на засобами мов високого рівня, практичного застосування існуючих систем управління базами даних; вживання ефективних моделей забезпечення даних на основі вивчення предметної області, методів аналізу, пошуку та використання існуючих систем управління базами даних; знайомство з існуючими системами управління базами даних реляційного та нереляційного типів; забезпечення теоретичної та інженерної підготовки фахівців у галузі проектування та реалізації бізнес-застосувань, що взаємодіють з системами управління базами даних.

Робоча програма навчальної дисципліни передбачає навчання у формі лекцій, лабораторних робіт та самостійної підготовки студентів. Для практичного засвоєння основних тем дисципліни лабораторні роботи проводяться із застосуванням комп’ютерів, локальних мереж та мережі Internet у комп’ютерних класах ХНЕУ.

Об'єктом навчальної дисципліни є інформаційні системи та процеси у певних предметних областях, що потребують для своєї обробки використання новітніх комп’ютерних технологій для зберігання, обробки та аналізу великих об'ємів даних.

Предметом навчальної дисципліни є безпосередньо сама база даних як сховище даних; базові моделі, що лежать і основі сучасних баз даних; мовні засоби спілкування проектувальників та користувачів з СКБД; засоби проектування баз даних та застосувань з ними; технології доступу до даних, що зберігаються в базах даних.

Вивчення навчальної дисципліни спрямовано на отримання студентами компетентностей у галузі розробки бізнес-застосувань з використанням сучасного програмного

забезпечення, що дозволить майбутнім фахівцям вирішувати складні економічні задачі у подальшій професійній діяльності.

Необхідним елементом успішного засвоєння навчального матеріалу дисципліни є самостійна робота студентів з літературою з питань проектування та використання баз даних, розробки та супроводу відповідних програмних продуктів.

Усі види занять розроблено відповідно до положень кредитно-модульної системи процесу навчання.

У результаті вивчення навчальної дисципліни студент повинен:

знати:

принципи побудови архітектура сучасних бізнес-застосувань;

основні концепції організації з'єднань зі сховищами даних в програмних продуктах;

логіку розвитку технологій доступу до даних;

концепції та принципи використання класичних засобів доступу до даних;

відмінності між з'єднаним й роз'єднаним середовищем в організації доступу до даних;

основні компоненти архітектура ADO.NET, їхнє призначення та взаємодію;

призначення класів, що входять до простору імен System.Data;

основні етапи алгоритму створення з'єднання із джерелом даних в програмних проектах;

основні засоби керування з'єднанням зі сховищем даних в програмних продуктах,

вимоги до застосувань, що одночасно обслуговують значну кількість клієнтів бази даних, та засоби їхньої програмної реалізації;

алгоритми виконання операцій у роз'єднаному середовищі та програмні засоби їхньої реалізації;

роль набору даних DataSet в організації автономної роботи з базою даних та його структура;

роль об'єктів прив'язки даних до інтерфейсу в організації колективної роботи над програмними проектами;

основні підходи до реалізації CRUD-операцій з ієрархічними даними;

призначення технології JDBC;

засоби створення з'єднання з базою даних засобами мови Java;

способи використання об'єкта ResultSet в програмних проектах мовою Java;

засоби отримання метаданих про запити і базу даних в цілому;

засоби реалізації концепції збережених процедур в програмних проектах мовою Java;

призначення й переваги типізованих наборів даних;

способи створення типізованих наборів даних та засоби роботи з ними;

методи автоматизації побудови інтерфейсу користувача на основі типізованих наборів даних;

призначення й переваги технології LINQ в розробці програмних продуктів;

концепції відкладених й негайних операції в LINQ;

призначення платформи Entity Framework та її сценарії;

моделі й основні поняття в Entity Framework;

призначення технології Code First та її переваги;

концепцію міграцій даних в Code First;

основні алгоритми побудови застосувань з використанням технології Code First;

основні засоби візуалізації даних в бізнес-застосуваннях;

поняття звіту в Visual Studio та засоби і методи створення.

вміти:

створювати N-рівневі програмні продукти;

формувати рядки підключень до баз даних різних типів;

використовувати класи провайдерів даних ADO.NET;

використовувати механізми роботи пулів з'єднань;
здійснювати програмну реалізацію CRUD-операцій з базами даних;
створювати набори даних DataSet;
використовувати класів адаптерів даних ADO.NET;
відображати дані у застосуваннях і виконувати операцій ведення бази даних у роз'єднаному середовищі;
завантажувати схеми таблиць із бази даних;
здійснювати програмну реалізацію спільного ведення батьківської й дочірньої таблиць;
створювати з'єднання з базою даних мовою Java;
реалізовувати сценарії виконання CRUD-операцій мовою Java;
отримувати метадані об'єкта ResultSet;
використовувати технології пакетних і підготованих запитів;
працювати зі збереженими процедурами засобами мови Java;
створювати типізовані набори даних;
змінювати властивості об'єктів у типізованих наборах даних;
використовувати адаптери таблиць;
відображати дані у застосуваннях на основі типізованих наборів даних;
формувати LINQ-запитів до DataSet;
групувати дані і виконувати обчислення агрегованих величин за технологією LINQ to DataSet;
здійснювати операції об'єднання даних кількох таблиць та створювати підзапити;
будувати діаграми із використанням технології LINQ to DataSet;
будувати класи сутностей та клас DbContext;
використовувати засобів Data Annotations та Fluent API для налаштування сутностей;
удосконалювати моделі даних на основі міграцій;
розробляти застосування Windows Forms на основі технології Code First;
створювати документи на основі даних, що зберігаються в базі;
здійснювати програмну реалізацію операцій фільтрування даних, що відображаються у звітах;
будувати програмні засоби бізнес-аналізу даних.

Отримані знання, вміння та навички дозволять працювати за професіями згідно з Національним класифікатором професій ДК 003:2010:

2131.2 Адміністратор бази даних;
2131.2 Адміністратор даних;
2131.2 Інженер з програмного забезпечення комп'ютерів;
2132.2 Інженер-програміст;
2132.2 Програміст (база даних);
2132.2 Програміст прикладний;
3121 Фахівець з розроблення комп'ютерних програм.

Обсяг дисципліни в кредитах ЄКТС та його розподіл у годинах за формами організації освітнього процесу та видами навчальних занять подано в табл. 1.1.

Таблиця 1.1

Обсяг дисципліни (в кредитах ЄКТС) та його розподіл за формами організації (у годинах)

Курс	**3**	
Семестр	**6**	
Кількість кредитів ECTS	**5**	
Аудиторні навчальні заняття	**лекції**	**16**
	лабораторні	**48**

Самостійна робота		**86**
Форма підсумкового контролю	**іспит**	

Вивчення дисципліни "Технології баз даних" ґрунтується на знаннях та вміннях, які студенти отримали під час вивчення дисциплін математичного циклу та циклу програмування. Водночас вона забезпечує базові та вибіркові дисципліни, що входять до циклів проектування інформаційних систем, аналізу даних і просунутого програмування. Структурно-логічна схема вивчення навчальної дисципліни "Технології баз даних" подано в табл. 1.2.

Таблиця 1.2

Структурно-логічна схема вивчення навчальної дисципліни

Попередні дисципліни	Наступні дисципліни
Вступ до комп'ютерних наук	Системний аналіз та проектування інформаційних систем
Математичний аналіз	Веб-технології та веб-дизайн
Дискретна математика	Моделювання інформаційних систем
Програмування	Захист інформації
Алгоритми та структури даних	Бази даних Data Mining
Основи об`єктно-орієнтованого програмування	Інтелектуальний аналіз даних
Бази даних	Програмування для мобільних пристроїв
Моделювання систем та методи оптимізації	Технології розробки та тестування програмного забезпечення
Операційні системи	Сучасні Java-технології
Комплексний курсовий проект: Програмування	Комплексний курсовий проект: Проектування

2. Компетентності та результати навчання за дисципліною:

У процесі викладання навчальної дисципліни основна увага приділяється оволодінню студентами професійними компетентностями, що наведені в табл. 2.1.

Таблиця 2.1

Структурно-логічна схема вивчення навчальної дисципліни

Компетентності	Результати навчання
Здатність використовувати режими з'єднаного та роз'єднаного середовища для доступу до даних в бізнес-застосуваннях	Володіти принципами побудови архітектури сучасних бізнес-застосувань
	Застосовувати основні концепції організації з'єднань зі сховищами даних в програмних продуктах
	Вибирати найбільш доцільне середовище організації доступу до даних
	Використовувати в проектах основні компоненти ADO.NET та JDBC
	Будувати алгоритми створення з'єднання із джерелом даних в програмних проектах
	Проектувати засоби керування з'єднанням зі сховищем даних в програмних продуктах

	Організовувати автономну роботу з базою даних
	Створювати програми для виконання операцій з даними у з'єднаному та роз'єднаному середовищі
	Прив'язувати дані до інтерфейсу користувача в програмних продуктах
Здатність застосовувати сучасні теорії організації баз даних, методи і технології їх розробки	Уміти використовувати переваги типізованих наборів даних під час програмування бізнес-застосувань
	Застосовувати технологію LINQ в задачах аналізу даних
	Визначати найбільш раціональні сценарії використання платформи Entity Framework
	Будувати моделі даних засобами технології Code First
	Виконувати налаштування сутностей в сценарії Code First
	Модифікувати моделі даних засобами міграцій
	Програмувати застосування Windows Forms на основі технології Code First
	Проектувати і створювати документи на основі даних, що зберігаються в базі
	розробляти програмні засоби бізнес-аналізу даних

3. Програма навчальної дисципліни

Теми лекцій

Змістовий модуль 1.

Класичні засоби доступу до даних

Тема 1. З'єднання з базами даних в ADO.NET. Виконання операцій у з'єднаному середовищі

1.1. Призначення ADO.NET.

Взаємодія застосування і даних. Локальні бази даних, розподілені бази даних підприємства та XML-сховища. ADO.NET як провідна технологія забезпечення зв’язку застосування і бази даних. Визначення ADO.NET. Трирівнева архітектуру сучасних застосувань.

1.2. Розвиток технологій доступу до даних.

ODBC API – технологія, що побудована на множині драйверів, які забезпечують доступ до конкретних СКБД. Надбудови DAO і RDO в технології ODBC. Забезпечення низькорівневого інтерфейсу у технології OLE DB. Спрощений доступ до роз’єднаних даних у технології ADO.NET. Подальший розвиток технологій доступу до даних в платформі Entity Framework.

1.3. З'єднане й роз'єднане середовище.

Алгоритм взаємодії застосування з базою даних. З’єднане та роз’єднане середовище як різновиди встановлення з’єднання застосування з базою даних. Пе-

реваги і недоліки з’єднаного та роз’єднаного середовищ, оптимальні сфери їхнього застосування.

1.4. Архітектура ADO.NET. Простір імен System Data.

Роль постачальника даних в реалізації технології доступу до даних. Постачальники даних ODBC, OLE DB, SQL Server, Oracle, EntityClient. Компоненти постачальника даних Connection, Command, DataReader, DataAdapter. Призначення об’єкта DataSet. Узагальнені та конкретні імена об’єктів.

Базова бібліотека доступу до даних System.Data.dll. Вміст просторів імен System.Data, System.Data.Common, System.Data.SqlClient, System.Data.OleDb, System.Data. OracleClient, System.Data.Odbc, System.Data.SqlTypes, System.Xml.

1.5. Створення з'єднання із джерелом даних. Побудова рядка підключення за допомогою майстра.

Алгоритмом роботи застосування, що використовує технологію доступу до даних ADO.NET та його відмінності між з’єднаним та роз’єднаним середовищем. Послідовність кроків в реалізації створення з'єднання з базою даних. Конструктор об'єкта Connection. ConnectionString як найголовніша властивість об'єкта Connection. Параметри властивості ConnectionString Provider, Driver, Connection Timeout, Initial Catalog, Data Source, Password, User ID, Integrated Security. Базовий формат рядка підключення. Найпростіші формати рядка підключення для різних баз даних.

Етапи роботи Майстра налаштування джерела даних в Visual Studio. Призначення фай конфігурації app.config.

1.6. Управління з'єднанням.

Підстановка |DataDirectory|. Переваги використання будівника рядків з'єднання Connection StringBuilder. Методи об’єкта Connection. Призначення блока using. Значення властивості State.

1.7. Пул з'єднань.

Призначення пулу з'єднань. Доцільність використання пулу з'єднань. Випадки, коли створюється новий пул з'єднань. Засоби для відмови від створення пулів з'єднань.

1.8. Організація роботи в з'єднаному середовищі. Створення і запуск командних об'єктів.

Виробничі ситуації, в яких необхідно використовувати з'єднане середовище. Наслідки не використання роз'єднаного середовища у цих ситуаціях. Ресурсозатратність організація оброблення даних у з'єднаному середовищі. Етапи, з яких складається процес оброблення даних у з'єднаному середовищі і дії, які виконуються на кожному з них.

Призначення має класу DBCommand. Властивості об’єкта DBCommand Connection, CommandType, CommandText, Parameters та методи ExecuteReader, ExecuteScalar, ExecuteNonQuery.

1.9. Клас DataReader.

Призначення класу DataReader. Преваги класу DataReader. Створення об’єкта DataReader. Функціональність методу Read. Методи, які дозволяють звертатися до значень стовпців за їхніми типами даних (GetDateTime, GetDouble, GetInt32 тощо). Організація обробки результаті запиту з кількома рядками. Необхідність закривання об’єкта DataReader. Обробка відразу декілька результуючих множин (по одній на кожен запит).

1.10. Операції CRUD.

Використання методу ExecuteNonQuery класу DBCommand для реалізації CRUD-операцій. Створення бази даних та її видалення. Створення таблиці. Додавання даних у таблицю (створення даних). Зміна і вилучення вибраних даних з таблиць.

Тема 2. Виконання операцій у роз'єднаному середовищі

2.1. Створення об’єктів для обробки даних роз'єднаному середовищі.

Задачі бізнесу, в яких доцільно використовувати роз'єднане середовище. Механізм організації автономної роботи застосування. Масштабованість оброблення даних у роз'єднаному середовищі. Основні етапи процесу оброблення даних у роз'єднаному середовищі.

2.2. Призначення й структура набору даних DataSet.

Роль класу DataSet в архітектурі ADO.NETдля роз'єднаного середовища. Призначення основних елементи об’єктної моделі DataSet. Колекція таблиць DataTable. Зв'язки між таблицями за допомогою об'єктів класу DataRelation. Забезпечення цілісності даних у класі DataSet за допомогою об'єктів UniqueConstraint і ForeignkeyConstraint. Методи DataSet (GetChanges, AcceptChanges, RejectChanges, Copy, Clone, Merge). Властивості DataSet (DataSetName, EnforceConstraints, Tables). Програмна реалізація алгоритму обробки даних у роз'єднаному середовищі.

2.3. Методи й властивості таблиць DataTable.

Створення об'єкта DataTable. Основні властивості класу DataTable (Columns, Rows, PrimaryKey, ChildRelations, ParentRelations, Constraints) та події (RowChanged, RowChanging, RowDeleting, RowDeleted).

2.4. Стовпці й рядки таблиці.

Призначення класу DataColumn. Основні властивості класу DataColumn (DataType, ColumnName, Caption, AllowdbNull, ReadOnly, Unique, AutoIncrement, AutoIncrementSeed, AutoIncrementStep, DefaultValue, Expression).

Призначення класу DataRow. Основні властивості класу DataRow (Item, RowState). Призначення методів Datatable.NewRow, DataTable.Add, DataTable.Delete, DataTable.AcceptChanges.

2.5. Зв'язки між таблицями.

Реляційна модель бази даних. Роль зв’язків між таблицями в DataSet. Призначення класу DataRelation. Створення відношення між локальними таблицями. Роль об'єкта ForeignKeyConstraint в контролі за змінами значень в дочірніх рядках. Реалізація відношень на складених ключах. Використання методів GetChildRows та GetParentRow. Посилання на стовпці в батьківській і дочірніх локальних таблицях.

2.6. Клас DataAdapter.

Призначення класу DataAdapter. Роль об'єкта Connection у класі DataAdapter. Методи Fill і Update. Алоритм завантаження даних з бази даних в локальну таблицю. Властивості адаптера SelectCommand, InsertCommand, DeleteCommand та UpdateCommand. Додавання рядка в локальну таблицю і збереження його в базі даних. Оновлення і видалення даних в локальній таблиці. Призначення об’єкта CommandBuilder.

2.7. Прив'язка даних до інтерфейсу.

Призначення класу BindingSource. Роль об’єктів прив'язки даних до інтерфейсу в організації колективної роботи над програмними проектами. Властивості з’єднувача DataSource, Sort і Filter. Методи BindingSource (MoveFirst, MoveLast, MovePrevious, MoveNext, AddNew, Insert, Remove, EndEdit). Властивість DataBindings як колекція об’єктів Binding. Використання об’єкта BindingSource під час створення багаторівневого застосування.

2.8. Виконання операцій видалення, додавання і модифікації ієрархічних даних.

Відношенням один-до-багатьох (master-detail). Ієрархічне оновлення даних. Основні підходи до реалізації CRUD-операцій з ієрархічними даними. Правила, що забезпечують збереження посилальної цілісності даних у базі. Ускладннення процесу додавання ієрархічних даних у разі, коли первинні ключі визначені із власти-

вістю автоінкременту. Алгоритм збереження ієрархічно пов'язаих даних.

Тема 3. JDBC – доступ до даних у Java

3.1. Призначення JDBC.

JDBC як платформо-незалежний промисловий стандарт взаємодії Java-застосувань з різними СКБД. Паки java.sql і javax.sql. Концепія драйверів. Динамічність завантаження драйверів. Типи драйверів JDBC-ODBC – міст, зовнішні бібліотеки СКБД, з проміжним Java-сервером, мережевий драйвер від виробника СКБД. Алгоритм виконання завдань за технологією JDBC.

3.2. СКБД Java DB (Derby).

Призначення СКБД Apache Derby та її обсяг. СКБД Apache Derby та Java DB. Історія розвитку Java DB. Середовища функціонування. Використання як вбудованої бази даних та з'єднання клієнт-сервер. Характеристики Java DB (підтримка мови SQL, типи даних та збережені процедури і функції). Обмеження в Java DB (таблиця як файл, розмір файла, кількість таблиць, результати тестувань).

3.3. З'єднання.

Імпорт JDBC пакетів. Реєстрація JDBC драйвера. Задавання URL бази даних. Створення об'єкта Connection. Закриття.

3.4. Виконання запитів.

Призначення класів Statement, PreparedStatement та CallableStatemen. Створення об'єкта Statement та його методи execute, executeUpdate і executeQuery. Етапи роботи з об'єктом класу PreparedStatement (створення параметричного запиту, компілювання запиту, задавання параметрів, виконання запиту). Пакетне виконання запитів.

3.5. ResultSet - результат виконання SELECT.

Типи поводження курсора. Узгодженість щодо читання і оновлення даних. Методи навігації записами ResultSet (beforeFirst, afterLast, last, moveToCurrentRow тощо). Оновлення ResultSet (зміна даних в самому ResultSet та збереження змін в базі даних).

3.6. Метадані.

Отримання даних про запит у класі ResultSetMetaData та даних про базу даних у класі DatabaseMetaData. Призначення методів getColumnCount, getColumnName та getColumnType класу ResultSetMetaData. Призначення методів getDatabaseProductName, getDatabaseProductVersion, getDriverName, getUserName, getURL та getTables класу DatabaseMetaData.

3.7. Збережені процедури.

Збережена процедура як група операторів, що виконує певне завдання на боці сервера бази даних. Вхідні та вихідні параметри. Особливості збережених процедур в Java DB. Основні кроки створення і використання збережених процедур.

Змістовий модуль 2.
Сучасні засоби доступу до даних

Тема 4. Типізовані набори даних

4.1. Переваги типізованих наборів даних.

Вбудована схема бази даних як засіб більш високої продуктивності і зменшення помилок етапу компілювання. Розширення класу DataSet. Уніфікація імена об'єктів. Доступність імен таблиць, стовпців та інших об'єктів через властивості. Швидка розробка інтерфейсу користувача методом drag-and-drop.

4.2. Способи створення типізованих наборів даних.

Майстер настроювання джерела даних. Конструктор наборів даних. Сфери застосування кожного способу створення типізованих наборів даних.

4.3. Робота з таблицями у вікні конструктора наборів даних.

Додавання стовпців в об’єкт DataTable (у тому числі з автоматичною генерацією значення). Налаштування властивостей об’єкта DataColumn (ім'я, заголовок, тип даних значення за замовчуванням, обмеження унікальності значень, встановлення первинного ключа, обчислюване значення тощо). Редагування зв’язків між таблицями.

4.4. Методи типізованих наборів даних.

Методи, які генерує Visual Studio для типізованих наборів даних. Спрощення введення тексту програми засобами IntelliSense. Додавання нового запису до типізованої таблиці. Пошук запису із заданим ключем. Редагування значення поля для заданого запису в типізованій таблиці.

4.5. Адаптери таблиць.

Індивідуальність адаптера таблиці в типізованому наборі даних. Адаптер таблиці як зв'язок між застосуванням і базою даних. Інкапсуляція запитів в адаптерах таблиць. Призначення методів Fill, Update і GetData. Створення додаткових запитів в адаптері таблиці.

4.6. Зв'язування з інтерфейсом користувача

Призначення вікна Data Sources в Visual Studio і його структура. Побудова інтерфейсу користувача засобами вікна Data Sources. Область компонентів у вікні конструктора форми Windows Forms. Налаштування виду елементів керування у вікні Data Sources. Побудова інтерфейсу master-detail.

Тема 5. Технологія LINQ to DataSet

5.1. Призначення й переваги LINQ.

LINQ як шаблони і технології для запиту даних. Використання засобів декларативної мови запитів для виконання операцій з об’єктами. Перенесення засобів роботи з реляційними базами даних мовою SQL у мову програмування. Зручність конструкцій запитів LINQ в мовах C# і Visual Basic у порівнянні з традиційнми запитами до даних. Зменшення помилок етапу компілювання. Покрокове виконання, встановлення точок зупинки, перегляд результатів у вікнах налагоджувальника.

5.2. Види технологій LINQ.

LINQ як родина технологій, що вбудовані у мови програмування C# та Visual Basic. Призначення технологій LINQ to Objects, LINQ to XML, LINQ to DataSet, LINQ to SQL, LINQ to Entities.

5.3. Технологія LINQ to Dataset і запити.

Розрив між засобами, за допомогою яких дані вибирають із бази, і тими, за допомогою яких дані вибирають з DataSet. Спрощення й прискорення написання запитів до даних в об'єкті DataSet на основі конструкції мови, які подібні до тих, що вживаються у мові SQL. Сфери використання технології LINQ to DataSet. Порядок роботи з LINQ до DataSet. Інформація, яку вказують у запитах LINQ to DataSet.

5.4. Вираз запиту.

Запит як набір інструкцій, що описують, які дані необхідно вибрати із зазначеного джерела даних, а також визначають форму й організацію даних, що вибирають. Запис запиту у формі виразу запиту (за допомогою синтаксису запиту). Речення запиту. Змінна запиту та її тип. Використання анонімного типу у запиті. Правила побудови запиту LINQ.

5.5. Речення запиту.

Призначення і синтаксис речень from, select (проекція), group, where, orderby, join, let. Ключове слово into. Вкладений запит.

5.6. Синтаксис методів.

Перетворення виразів запитів для джерела даних у виклики методів. Спосіб запису запиту у формі виклику методів. Застосування лямбда-виразів у синтаксисі методів. Порівняння синтаксису виразів запитів із синтаксисом методів. Випадки,

коли можна використовувати тільки синтаксис методів.

5.7. Відкладені й негайні операції.

Визначення відкладених та негайних операцій. Умови негайного виконання запиту LINQ. Умови выдкладеного виконання запиту LINQ.

5.8. Типізовані DataSet.

Різниця між запитами LINQ до типізованих і нетипізованих об’єктів DataSet. Переваги іменування, які надають типізовані об'єкт DataSet під час запису запиту LINQ. Запит LINQ to DataSet надає Доступ до значень стовпців запиту LINQ.

Тема 6. Платформа Entity Framework. Технологія Code First

6.1. Призначення платформи EntityFramework.

Об'єднання концепцій реляційної бази даних і об'єктно орієнтованого програмування в одну як мета платформи Entity Framework. Різниця між технологією типізованого набору даних і платформою Entity Framework. Сфери використання технологій типізованого набору даних і платформою Entity Framework.

6.2. Моделі і основні поняття в EntityFramework.

Сутнісну модель даних EDM (Entity Data Model). Концептуальна модель (у програмі). Модель збереження даних (у базі даних). Відображення об’єктів концептуальної моделі на об’єкти схеми збереження даних. Мови CSDL, SSDL та MSL. Відповідність термінів, що використовують в концептуальній і фізичній моделях даних.

6.3. Сценарії створення моделі EDM.

Сфери використання сценаріїв Model First, DB First, та Code First. Алгоритм побудови моделі EDM і бази даних за сценарієм Model First. Алгоритм побудови моделі EDM за сценарієм DB First. Імена сутностей в однині і множині.

6.4. Операції CRUD.

Призначення об’єкта класу ObjectContext і його склад. Призначення методу SaveChanges класу ObjectContext. Читання даних з бази даних засобами Entity Framework. Додавання даних в базу даних. Зміна даних в базі даних. Видалення даних з бази даних. Відмінності між виконанням операцій CRUD з використанням технології типізованих наборів даних і платформи Entity Framework

6.5. LINQ to Entities.

Підтримка технології LINQ у запитах до сутностей на основі LINQ to Entities. Використання у LINQ to Entities синтаксису виразів запитів і синтаксису запитів на основі методів. Реалізація операцій отримання в запиті не всіх властивості сутностей, а також результатів обчислень з їхніми властивостями. Фільтрація та сортування даних в запиті. Виконання агрегатних операторів Count, Sum, Average, Min та Max. Реалізація секціонування як засобу для виведення даних із довгих списків на web-сторінку. Навігація за зв'язками між сутностями. З'єднання як операція у запитах до джерел даних, що не мають доступних для переходу зв'язків один з одним за властивостями навігації.

6.6. Відображення даних.

Способи відображення в елементах керування на формі даних, що відібрані в запиті. Запит LINQ як джерело даних елемента керування. Запит LINQ як джерело даних з’єднувача. Візуальні засоби на основі панелі Data Sources.

6.7. Призначення технології Code First та її переваги.

Ситуації, в яких використовують технологію Code First. Розмір коду. Заміна класів класи ObjectContext і ObjectSet. Формування ”у польоті” моделі EDMX. Роль міграцій у плавній зміні бази даних. Використання візуальних засобів побудови інтерфейсу користувача.

6.8. Класи сутностей. Об’єкти доступу до даних.

Опис предметної області в коді за допомогою класів сутностей. Класи POCO (Plain Old CLR Objects). Відображення зв’язків між таблицями за допомогою влас-

тивостей навігації. Батьківські і дочірні сутності. Однина і множина імен сутностей. Імена ключових властивостей. Реалізація відкладеного завантаження за допомогою модифікатора virtual. Доступ до бази даних з використанням класу контексту.

6.9. Бібліотеки EntityFramework.

Призначення бібліотек EntityFramework.dll та EntityFramework.SqlServer.dll. Майстер NuGet PackageManager. Додавання майстра NuGet до попередніх версій Visual Studio. Алгоритм завантаження в проект бібліотеки EntityFramework.dll та EntityFramework.SqlServer.dll за допомогою майстра NuGet. Спільне використання бібліотек EntityFramework.dll та EntityFramework.SqlServer.dll у кількох проектах рішення.

6.10. Створення бази даних.

Сервери СКБД SQL Server, на яких автоматично створюється база даних у сценарії Code First. Ім'я бази даних, що створюється автоматично. Установлення наперед заданого імені бази даних. Необхідна умова створення бази даних під час першого виконання застосування. Задавання стратегії роботи з базою даних в застосуванні.

6.11. Домовленості в Code First.

Ім'я бази даних. Ім'я таблиці. Кількість таблиць. Первинний ключ. Зовнішній ключ. Кратність відношення. Каскадне видалення. Типи даних стовпців.

6.12. Налаштування моделі даних засобами Data Annotations та Fluent API.

Простори імен, які потрібно використовувати для застосування засобів Data Annotations. Формат запису анотацій даних. Анотації даних для налаштування імені таблиці, первинного ключа, зовнішнього ключа, заборони Null, довжини рядка, заборони відображення, імені поля. Відмінності функціональності Fluent API від Data Annotation. Порядок налаштування відображення за допомогою Fluent API. Методи EntityTypeConfiguration, які найчастіше використовують для налаштування відображення між концептуальною моделлю і базою даних засобами Fluent API.

6.13. Удосконалення моделі (міграції).

Призначення міграцій в CodeFirst. Порівняння міграції і FluentAPI. Вікно Package Manager Console у керуванні міграціями. Команда для початку міграцій Enable-Migrations і файли InitialCreate.cs та Configuration.cs, Методи Up() і Down(). Формування шаблона міграції, що відповідає змінам в концептуальній моделі (команда Add-Migration). Виконання міграцій (команда Update-Database). Перехід до заданої міграції. Реєстрація ініціалізатора бази даних.

6.14. Побудова застосування з використанням технології Code First.

Призначення методів Load і ToBindingList та властивості Local. Забезпечення двосторонньої прив'язки та сортування для подачі даних ієрархічно пов'язаних таблиць. Клас ObservableListSource, що реалізує IListSource для колекцій.

7. Побудова звітів (Reporting)

7.1. Поняття звіту.

Призначення звітів. Звіти як друкований документ, що містить дані з бази даних. Структура звіту. Види звітів.

7.2. Засоби Visual Studio для роботи зі звітами.

Конструктор звітів. Елемент керування для відображення звіту ReportViewer. Технологія роботи зі звітами в Visual Studio. Робота елемента керування ReportViewer в режимі локальної обробки або в режимі віддаленої обробки. Структура вікна ReportViewer (схема документа, перехід сторінками, зупинка і оновлення звіту, друк звіту, зміна масштабу сторінки звіту, параметри сторінки, експорт звіту, пошук у звіті). Елементи керування в конструкторі звітів.

7.3. Таблікс (таблиці, матриці і списки).

Склад групи елементів таблікс і їхнє призначення.

7.4. Джерела даних.

Вікно властивостей елементів таблікс. Об'єкти DataTable та колекції бізнес-

об'єктов IEnumerable як як джерело даних звіту. Зв’язок полів звіту і бази даних. Вікно Report Data. Додавання нових таблиць в типізований DataSet на основі даних з інших локальних таблиць.

7.5. Створення звітів.

Створення нового RDLC-файла в майстрі звітів. Створення нового RDLC-файла в конструкторі звітів. Програмне формування структури звіту. Основні сценарії початку побудови звіту. Задавання даних звіту, розміщення полів, вибір макету та вибір стилю в майстрі звітів.

7.6. Схема документа.

Призначення елемента Схема документа у звіті. Сценарій додавання схеми документа до звіту. Керування розміром схеми документа під час побудови і перегляду звіту.

7.7. Фільтрація даних.

Призначення фільтрації даних у звіті. Способи фільтрації даних: спочатку завантажують всі дані таблиці, а потім фільтрують в DataSet і на сервері відбирають потрібні дані і тільки їх передають в DataSet та їхня програмна реалізація.

7.8. Параметри звіту.

Призначення параметрів звіту. Створення параметра та його використання. Програмна реалізація функціонування параметрів звіту.

7.9. Створення діаграм.

Призначення елемента керування Chart у вікні Report Designer. Налаштування властивостей елемента Chart. Побудова Dashboard в бізнес-застосуваннях.

Тема 8. Перспективи розвитку баз даних та технологій доступу до них.

8.1. Хмарні технології.

Переваги хмарних сховищ даних. Загальнодоступні та приватні сховища даних. Гібридні "хмари". Хмарні сховища Microsoft SQL Azure, Amazon Relational Database Service та Google Cloud Storage.

8.2. Бази даних NoSQL

Сфери використання баз даних SQL і NoSQL. Сховище «ключ-значення». Bigtable-подібні бази даних. Документо-орієнтоване сховище. Графова база даних. Структури даних і їх типи. Запити. Масштабованість. Надійність. Підтримка. Зберігання та доступ до складних структур даних. Сфери використання та можливості баз даних MongoDB. Доступ до даних, що зберігаються у базах даних NoSQL. Розробка застосувань з MongoDB.

8.3. Технології Hadoop і Apache Spark в Big Data.

Поняття великих даних. Набір ознак VVV (volume, velocity, variety). Джерела великих даних. Технології роботи з великими даними. Призначення технології Hadoop. Розподілена файлова система HDFS. Технології обробки даних MapReduce та YARN. Apache Spark як технологія для виконання швидких обчислень на кластерах. Програмні інтерфейси для мов Java, Scala, Python, R.

8.4. Останні версії SQL Server.

Версії SQL Server і їхні маркетингові імена. Рівні продукту. Основні нововведення. Кросплатформова версія SQL Server VNext. SQL Server Developer Edition для розробки та тестування. Підтримка і розвиток середовищ програмування. Регулярне оновлення середовища розробки SSMS.

8.5. Технологія PolyBase.

PolyBase як спрощення інтерфейсу користувача SQL Server з Hadoop та SQL Azure. Переваги PolyBase. Можливості PolyBase в SQL Server 2019. Розвертання і налаштування Hadoop для PolyBase. Під'єднання PolyBase до Hadoop як джерела даних. Обробка даних з використанням PolyBase. Масштабна група PolyBase. Зовнішнє джерело даних. Формат зовнішнього файлу. Ззовнішня таблиця. Отримання даних із кластера Hadoop. Імпорт даних в локальну таблицю. Вико-

ристання технології Spark в PolyBase.

8.6. Аналіз даних засобами SQL Server Machine Learning Services.

Убудовані в SQL Server засоби аналізу даних. Можливості мов R і Python у вирішенні задач аналізу даних. Переваги вмонтованих компонентів аналізу даних. Встановлення мов R і Pythonяк компонентів SQL Server. Запуск на виконання скриптів мовами R та Python у середовищі SQL Server Management Studio. Додавання підтримки для R, Python та Java в SQL Server 2019 для Linux.

8.7. Дослідження Gartner в галузі баз даних.

Дослідження ринків інформаційних технологій. Магічні квадранти компанії Gartner. Дослідження компанії Gartner в сфері баз даних. Операційні бази даних. Мережеві бази даних. Бази даних Transanlytical.

Теми лабораторних робіт

Лабораторна робота 1. Розробка програм виконання операцій у з'єднаному середовищі.

Лабораторна робота 2. Розробка програм виконання операцій у роз'єднаному середовищі.

Лабораторна робота 3. Розробка програм на основі інтерфейсу JDBC.

Лабораторна робота 4. Розробка програм з використанням типізованих наборів даних.

Лабораторна робота 5. Розробка програм з використанням технології LINQ to DataSet.

Лабораторна робота 6. Реалізація доступу до даних на основі технології Code First.

Лабораторна робота 7. Реалізація звітів в бізнес-застосуваннях.

4. Порядок оцінювання результатів навчання

Система оцінювання сформованих компетентностей у студентів враховує види занять, які згідно з програмою навчальної дисципліни передбачають лекційні, лабораторні заняття, а також виконання самостійної роботи. Оцінювання сформованих компетентностей у студентів здійснюється за накопичувальною 100-бальною системою. Відповідно до Тимчасового положення "Про порядок оцінювання результатів навчання студентів за накопичувальною бально-рейтинговою системою" ХНЕУ ім. С. Кузнеця, контрольні заходи включають:

поточний контроль, що здійснюється протягом семестру під час проведення лекційних та лабораторних занять і оцінюється сумою набраних балів (максимальна сума – 60 балів; мінімальна сума, що дозволяє студенту складати іспит, – 35 балів);

модульний контроль, що проводиться у формі колоквіуму як проміжний мініекзамен з ініціативи викладача з урахуванням поточного контролю за відповідний змістовий модуль і має на меті інтегровану оцінку результатів навчання студента після вивчення матеріалу з логічно завершеної частини дисципліни – змістового модуля;

підсумковий/семестровий контроль, що проводиться у формі семестрового екзамену, відповідно до графіку навчального процесу.

Порядок проведення поточного оцінювання знань студентів. Оцінювання знань студента під час лабораторних занять та виконання індивідуальних завдань

проводиться за такими критеріями:

розуміння, ступінь засвоєння теорії та методології проблем, що розглядаються; ступінь засвоєння фактичного матеріалу навчальної дисципліни; ознайомлення з рекомендованою літературою, а також із сучасною літературою з питань, що розглядаються; вміння поєднувати теорію з практикою при розгляді виробничих ситуацій, розв'язанні задач, у процесі виконання індивідуальних завдань та завдань, винесених на розгляд в аудиторії; логіка, структура, стиль викладу матеріалу в письмових роботах і при виступах в аудиторії, вміння обґрунтовувати свою позицію, здійснювати узагальнення інформації та робити висновки; здатність проводити критичну та незалежну оцінку певних проблемних питань; вміння пояснювати альтернативні погляди та наявність власної точки зору, позиції на певне проблемне питання; застосування аналітичних підходів; якість і чіткість викладення міркувань; логіка, структуризація та обґрунтованість висновків щодо конкретної проблеми; самостійність виконання роботи; грамотність подачі матеріалу; використання методів порівняння, узагальнення понять та явищ; оформлення роботи.

Загальними критеріями, за якими здійснюється оцінювання позааудиторної самостійної роботи студентів, є: глибина і міцність знань, рівень мислення, вміння систематизувати знання за окремими темами, вміння робити обґрунтовані висновки, володіння категорійним апаратом, навички і прийоми виконання практичних завдань, вміння знаходити необхідну інформацію, здійснювати її систематизацію та обробку, самореалізація на лабораторних заняттях.

Підсумковий контроль знань та компетентностей студентів з навчальної дисципліни здійснюється на підставі проведення семестрового екзамену, завданням якого є перевірка розуміння студентом програмного матеріалу в цілому, логіки та взаємозв'язків між окремими розділами, здатності творчого використання накопичених знань, вміння формулювати своє ставлення до певної проблеми навчальної дисципліни тощо.

Екзаменаційний білет охоплює програму дисципліни і передбачає визначення рівня знань та ступеня опанування студентами компетентностей.

Кожен екзаменаційний білет складається із 3 практичних ситуацій (стереотипне, діагностичне та евристичне завдання), які передбачають вирішення типових професійних завдань фахівця на робочому місці та дозволяють діагностувати рівень теоретичної підготовки студента і рівень його компетентності з навчальної дисципліни.

Результат семестрового екзамену оцінюється в балах (максимальна кількість – 40 балів, мінімальна кількість, що зараховується, – 25 балів) і проставляється у відповідній графі екзаменаційної "Відомості обліку успішності".

Студента слід вважати атестованим, якщо сума балів, одержаних за результатами підсумкової/семестрової перевірки успішності, дорівнює або перевищує 60. Мінімально можлива кількість балів за поточний і модульний контроль упродовж семестру – 35 та мінімально можлива кількість балів, набраних на екзамені, – 25.

Підсумкова оцінка з навчальної дисципліни розраховується з урахуванням балів, отриманих під час екзамену, та балів, отриманих під час поточного контролю за накопичувальною системою. Сумарний результат у балах за семестр складає: "60 і більше балів – зараховано", "59 і менше балів – не зараховано" та заноситься у залікову "Відомість обліку успішності" навчальної дисципліни.

Шкала оцінювання: національна та ЄКТС

Сума балів за всі	Оцінка	Оцінка за національною шкалою

види навчальної діяльності	ЄКТС	для екзамену, курсового проекту (роботи), практики	для заліку
90 – 100	A	відмінно	зараховано
82 – 89	B	добре	
74 – 81	C		
64 – 73	D	задовільно	
60 – 63	E		
35 – 59	FX	незадовільно	не зараховано
1 – 34	F		

Розподіл балів за тижнями

Теми змістового модуля			**Лекційні заняття**	**Лабораторні заняття**	**Захист лабораторної роботи**	**Письмова контрольна робота**	**Усього**
Змістовий модуль 1. Класичні засоби доступу до даних	**Тема 1.** З'єднання з базами даних в ADO.NET. Виконання операцій у з'єднаному середовищі	1 тиждень	1,5	0,5			2
		2 тиждень		1	4		5
	Тема 2. Виконання операцій у роз'єднаному середовищі	3 тиждень	1,5	0,5			2
		4 тиждень		1	4		5
	Тема 3. JDBC – доступ до даних у Java	5 тиждень	1,5	0,5			2
		6 тиждень		1	4	4	9

Змістовий модуль 2. Сучасні засоби доступу до даних	**Тема 4.** Типізовані набори даних	7 тиждень	1,5	0,5			2
		8 тиждень		1	4		5
	Тема 5. Технологія LINQ to DataSet	9 тиждень	1,5	0,5			2
		10 тиждень		1	4		5
	Тема 6. Платформа Entity Framework. Технологія Code First	11 тиждень	1,5	0,5			2
		12 тиждень		1	4		5
	Тема 7. Побудова звітів (Reporting)	13 тиждень	1,5	0,5			2
		14 тиждень		1	4		5
	Тема 8. Перспективи розвитку баз даних та технологій доступу до них	15 тиждень	1,5	1,5		4	7
	Іспит						40
Усього			12	12	28	8	100

5. Рекомендована література

Основна

1. Лосєв М. Ю. Організація баз даних та знань (ADO.NET) : конспект лекцій / М. Ю. Лосєв, О. В. Тарасов, В. В. Федько. – Х. : Вид. ХНЕУ, 201. – 108 с.

2. Федько В. В. Організація баз даних та знань : навч.-практ. посібн. / В. В. Федько, О. В. Тарасов, М. Ю. Лосєв. – Х. : Вид. ХНЕУ, 2013. – 200 с.

3. Федько В. В. Сучасні засоби доступу до даних : навч. посібн. для самостійної роботи студентів з навч. дисципліни «Організація баз даних та знань» / В. В. Федько, О. В. Тарасов, М. Ю. Лосєв. – Х. : Вид. ХНЕУ ім. С. Кузнеця, 2014. – 328 с.

4. Федько В. В. Классические средства доступа к данным: учебное пособие по учеб-ной дисциплине «Базы данных» для иностранных студентов : Мультимедийное интерактивное электронное издание комбинированного использования / В. В. Федько, А. В. Тарасов, М. Ю. Лосев. – Х. : Изд. ХНЭУ им. С. Кузнеца, 2016. – 218 с.

Додаткова

5. Лосєв М. Ю. Бази даних: навчальний посібник для самостійної роботи студентів / Лосєв М. Ю., Федько В. В. – Х. : Вид. ХНЕУ, 2018. – 225с.

6 Федько В. В. Лабораторний практикум з модуля "Основи баз даних та знань" навчальної дисципліни "Організація баз даних та знань" / В. В. Федько, О. В. Тарасов, М. Ю. Лосєв. – Х. : Вид. ХНЕУ, 2011. – 192 с.5.

7. Тарасов А. В. Лабораторный практикум для модуля "Моделирование

данных" учебной дисциплины "Организация баз данных и знаний" : электронное учебно-практическое пособие / А. В. Тарасов, В. В. Федько, М. Ю. Лосев. – Х. : Изд. ХНЭУ им. С. Кузнеца, 2014.

8. Нейгел К. C# 4.0 и платформа .NET 4 для профессионалов / К. Нейгел, Б. Ивьен, Дж. Глинн, К. Уотсон. – М. : ООО ИД «Вильямс», 2011. – 1440 с.

9. Тарасов О. В. Використання мови SQL для роботи з сучасними системами керування базами даних. Практикум з навчальної дисципліни "Організація баз даних та знань" : навч. посібн. / О. В. Тарасов, В. В. Федько, М. Ю. Лосєв. – Х. : Вид. ХНЕУ, 2013. – 349 с.

10. Тарасов О. В. Проектування баз даних : навч. посібн. / О. В. Тарасов, В. В. Федько, М. Ю. Лосєв. – Х. : Вид. ХНЕУ, 2011. – 200 с.

11. Johnson G. Exam 70-516: TS: Accessing Data with Microsoft .NET Framework 4 / G. Johnson. – Microsoft Press. 2011. – 671 p.

Інформаційні ресурси в Інтернеті

12. .NET Development. Available at: https://msdn.microsoft.com/en-us/library/ff361664(v=vs.110).aspx.

13. Develop Windows desktop applications. Available at: https://docs.microsoft.com/en-us/windows/deskto.

14. Oracle Application Development. – https://www.oracle.com/database/technologies/application-development.html.

15. ADO.NET. Available at: https://docs.microsoft.com/en-us/dotnet/framework/data/adonet.

16. Entity Framework DocumentationІнформаційні ресурси в Інтернеті. Available at: https://docs.microsoft.com/en-us/ef/index.

17. Get started with Entity Framework 6. Available at: https://docs.microsoft.com/en-us/ef/ef6/get-started.