

CPSC 304 Project Cover Page

Milestone #: 4

Date: November 30th, 2023

Group Number: 134

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Benny Li	28772598	cli66 e0p8y	cli66@student.ubc.ca
Helen Zhou	46073292	hzhou2 a2v5h	helenzhou212@gmail.com
Kelly Chen	36916120	chenkai y7v1z	kk.chen.1058@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above.

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Table of Contents

Project File Information	2
Project Description	3
Changes to ER Diagram, Schema, & SQL Script	4
Final Schema & Screenshots	5
SQL Queries, Locations, & Screenshots	12
1. INSERT Operation	12
2. DELETE Operation	15
3. UPDATE Operation	18
4. Selection	20
5. Projection	21
6. Join	22
7. Aggregation with Group By	24
8. Aggregation with Having	25
9. Nested Aggregation with Group By	26
10. Division	27
Appendices	29
Appendix A - SQL Script	29
Appendix B - ER Diagram	40
Appendix C - Schema Screenshots from SQL Plus	41

Project File Information

Our project repository can be accessed at the link below:

https://github.students.cs.ubc.ca/CPSC304-2023W-T1/project_a2v5h_e0p8y_y7v1z

Our full SQL script for setting up the database can be found inside the [/database](#) folder. A copy of the script is also included in Appendix A of this document for ease of access.

A copy of our project ER diagram can be found in Appendix B of this document.

A README.md file is included in our repository which provides the setup instructions for using PHP and Oracle as well as necessary permission changes for our own development team's ease of access. For the grader, there is no additional information to be provided other than the required deliverables, so we have included a README.txt file that says "No extra information" as per grading rubrics which can be accessed [here](#).

Project Description

The pumpkin patch application is an agricultural management tool to track and manage the necessary aspects of a patch, including pumpkin variety, visitor management, and marketing events. The application can be used by owners, farmers, visitors, government and regulatory bodies, educational institutions, etc. to manage pumpkin patches across the region. In our final project, we provided an intuitive frontend for patch owners and guests to interact with our database.

As a patch owner, the user is able to:

- Create New Activity
 - Add new children/adult activities to an existing pumpkin patch
- Filter Pumpkin Patch
 - Search & filter for valid pumpkin patches that meet a customizable list of constraints
- Update Pumpkin Patch
 - Update the information of an existing pumpkin patch from the database
- Delete Pumpkin Patch
 - Delete an existing pumpkin patch from the database

As a guest, the user is able to:

- View Pumpkin Patch Projections
 - View information stored in any attribute of any table from our existing database
- Find Patches by Minimum Activities
 - Find pumpkin patches that feature more than a given number of activities
- Find Patches by Minimum Map Region Size
 - Find pumpkin patches that contain regions larger than a given size
- Count Special Events
 - Select an existing pumpkin patch and show the number of events that it hosts
- View Patches Planting All Varieties
 - Search for pumpkin patches that plant all of the tracked pumpkin varieties
- View Average Activity Fees
 - View the average activity fees for all existing pumpkin patches

Changes to ER Diagram, Schema, & SQL Script

No changes were made to our ER diagram since our Milestone 2 submission.

No changes were made to our schema since the normalized version from our Milestone 2 submission.

Minor changes were made to the DDL statements for creating tables in the SQL Script since our Milestone 2 submission in order to better represent our schema & enforce the existing constraints from our ER diagram. These changes are detailed below.

- MapRegion: Updated the primary key to include both MapID and RegionID as MapRegion is a weak entity of PatchMap.
- PatchMap: Added UNIQUE constraint to PatchID to enforce the one-to-one relationship between PatchMap and PumpkinPatch.
- EquipmentLog: Added UNIQUE constraint to PatchID to enforce the one-to-one relationship between EquipmentLog and PumpkinPatch.
- Tickets: Updated the primary key to include both TicketID and EventID as Tickets is a weak entity of SpecialEvent.
- Length of VARCHAR2 types were reduced at various locations to enable schema screenshots from SQL Plus to fit into the document page.

Final Schema & Screenshots

We are providing screenshot for our schema through the frontend interface we implemented as a part of this project as our layout is much more intuitive & space-efficient compared to the SQL Plus interface. We have also included screenshots from the SQL Plus interface in Appendix C in case it is required for grading purpose.

- PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)

Patchid	Patchownership	Patchsize	Patchaddress	Patchname
1	Owner1	100	Address1	Sunny Farm
2	Owner2	200	Address2	Happy Farm
3	Owner3	150	Address3	Green Farm
4	Owner4	300	Address4	Organic Farm
5	Owner5	250	Address5	Country Farm

- PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)

Varietyid	Quantityplanted	Planteddate	Varietyname
1	50	01-APR-23	Big Max
2	75	15-APR-23	Sugar Pie
3	65	20-APR-23	Cinderella
4	80	01-MAY-23	Jarrahdale
5	100	10-MAY-23	Kabocha
6	100	10-MAY-23	Something Else

- PatchMap (MapID, **PatchID**, Paths, MapDescription)

Mapid	Patchid	Paths	Mapdescription
1	1	Path1	Description1
2	2	Path2	Description2
3	3	Path3	Description3
4	4	Path4	Description4
5	5	Path5	Description5

- MapRegion (RegionID, MapID, VarietyID, RegionSize)

Regionid	Mapid	Varietyid	Regionsize
1	1	1	20
2	1	2	10
3	1	3	5
4	1	4	2
5	1	5	18
6	2	1	25
7	2	2	25
8	2	3	25
9	2	4	25
10	2	5	25
11	3	1	30
12	3	2	15
14	3	3	25
13	3	4	7
15	3	5	1
16	4	1	50
17	4	2	35
18	4	3	28
19	4	4	17
20	4	5	44
21	5	1	75
22	5	2	2
23	5	3	60
24	5	4	75
25	5	5	80

- HarvestSchedule (PlantedDate, VarietyName, HarvestDate)

Planteddate	Varietyname	Harvestdate
01-APR-23	Big Max	01-OCT-23
15-APR-23	Sugar Pie	15-OCT-23
20-APR-23	Cinderella	20-OCT-23
01-MAY-23	Jarrahdale	01-NOV-23
10-MAY-23	Kabocha	10-NOV-23

- PatchTracksVariety (PatchID, VarietyID)

Patchid	Varietyid
1	1
1	2
1	3
1	4
1	5
1	6
2	2
2	3
2	4
3	3
4	4
5	1
5	2
5	3
5	4
5	5
5	6

- MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate)

Lastmaintenancedate	Nextmaintenancedate
01-MAR-23	01-SEP-23
15-MAR-23	15-SEP-23
01-APR-23	01-OCT-23
15-APR-23	15-OCT-23
01-MAY-23	01-NOV-23

- EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, **PatchID**)

Logid	Lastmaintenancedate	Equipmentcount	Patchid
1	01-MAR-23	5	1
2	15-MAR-23	7	2
3	01-APR-23	6	3
4	15-APR-23	8	4
5	01-MAY-23	10	5

- MarketingPlan (PlanName, PlanDescription, SocialMediaRecords, AdvertisingRecords, **PatchID**)

Planname	Plandescription	Socialmediarecords	Advertisingrecords	Patchid
Plan1	Description1	Record1	AdRecord1	1
Plan21	Description21	Record21	AdRecord21	2
Plan22	Description22	Record22	AdRecord22	2
Plan31	Description31	Record31	AdRecord31	3
Plan32	Description32	Record32	AdRecord32	3
Plan33	Description33	Record33	AdRecord33	3
Plan4	Description4	Record4	AdRecord4	4
Plan5	Description5	Record5	AdRecord5	5

- SpecialEvent (EventID, EventName, PlanName)

Eventid	Eventname	Planname
1	Halloween Bash	Plan1
211	Harvest Festival1,T	Plan21
212	Harvest Festival2	Plan21
221	Harvest Festival3	Plan22
222	Harvest Festival4	Plan22
31	Pumpkin Carving Contest1	Plan31
32	Pumpkin Carving Contest2	Plan32
33	Pumpkin Carving Contest3	Plan33
4	Thanksgiving Sale	Plan4

- Tickets (TicketID, EventID, AdmissionType, VisitorType)

Ticketid	Eventid	Admissiontype	Visitortype
1	1	Standard	Adult
2	211	Premium	Child
3	31	Standard	Senior
4	4	Standard	Adult
5	221	Premium	Adult

- Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName, **PatchID**)

Activityid	Duration	Fee	Activitydescription	Activityname	Patchid
1	30	5	Pumpkin carving activity	Carving Class	1
2	60	10	Pumpkin pie making activity	Cooking Class	2
3	45	7	Hayrides around the patch	Hayride	3
4	120	15	Halloween movie night	Movie Night	4
5	90	12	Pumpkin painting activity	Painting Class	5
6	90	20	Youth art class	Youth Art Class	1
7	120	50	Wine tasting event	Winery	1
8	60	10	Cultural dance showcase	Dance Fiesta	1
9	150	60	Cooking class for adults	Culinary Masters	1
10	120	25	Nighttime nature walk	Nature Walk	2
11	30	8	Pumpkin seed roasting	Seed Roasting	3
12	180	30	Fall photography class	Photography	4
13	75	18	Pumpkin-themed story hour	Story Hour	5
14	50	10	Scarecrow making workshop	Scarecrow Workshop	2
15	60	15	Fall festival preparation	Festival Prep	3
16	45	5	Pumpkin trivia contest	Trivia Contest	1
17	120	22	Farm-to-table cooking demo	Cooking Demo	4
18	30	6	DIY pumpkin spice candles	Candle Making	5

- KidsActivities (**ActivityID**, Guardian Requirement)

Activityid	Guardianrequirement
1	1
3	1
5	1
6	0
7	2
8	1
9	0
13	0
14	0
18	1

- AdultActivities (**ActivityID**, Age Requirement, Alcohol Involvement)

Activityid	Agerequirement	Alcoholinvolvement
2	21	0
4	21	1
6	0	0
7	0	1
8	0	0
9	0	1
10	18	1
11	18	0
12	20	0
14	0	0
15	20	1
16	22	1
17	23	1
18	0	0

SQL Queries, Locations, & Screenshots

Each of the sections below correspond to query rubric items in Milestone 4 grading criteria. Each section will include the query itself, a description of the query, location of the query within our project repository, and screenshots from our GUI demonstrating sample usage of the query.

1. INSERT Operation

Our insert operation query is demonstrated by allowing patch owners to insert a new activity into the Activities table. When a new activity is created, our system automatically generates a unique ActivityID while user is able to define all other fields. The user is mandated to input whether the new activity is a kid's activity or adult activity or both, which enforces the total-overlapping ISA relationship detailed in our ER diagram.

The main query is shown below. The implementation of this query as well as subsequent insert queries for the Kids Activities and Adult Activities table can be found in line 23-71 of the file located at /pages/create_activity.php.

```
INSERT INTO Activities
(ActivityID, PatchID, ActivityName, Duration, Fee, ActivityDescription)
VALUES
(ACTIVITIES_SEQ.NEXTVAL, :patchID, :activityName, :duration, :fee, :activityDescription)
RETURNING ActivityID INTO :activityID;
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

Our GUI doesn't display the existing Activities tables when prompting the user for entry. However, you can see from screenshots in the previous section that our initial database contains 18 entries in the Activities table & 10 entries in the Kids Activities table prior to adding a new activity.



Create Activity

Patch ID:

1

Activity Name:

new activity

Duration (in minutes):

10

Fee:

20

Description:

a new activity

☒ This is a kids activity

Guardian Requirement:

d

☐ This is an adults activity

Create Activity

AFTER:

New activity with ID 19 is inserted into both the Activities table and Kids Activities table.

New activity created successfully with ID: 19.

Activity Details:

Activityid	Duration	Fee	Activitydescription	Activityname	Patchid
1	30	5	Pumpkin carving activity	Carving Class	1
2	60	10	Pumpkin pie making activity	Cooking Class	2
3	45	7	Hayrides around the patch	Hayride	3
4	120	15	Halloween movie night	Movie Night	4
5	90	12	Pumpkin painting activity	Painting Class	5
6	90	20	Youth art class	Youth Art Class	1
7	120	50	Wine tasting event	Winery	1
8	60	10	Cultural dance showcase	Dance Fiesta	1
9	150	60	Cooking class for adults	Culinary Masters	1
10	120	25	Nighttime nature walk	Nature Walk	2
11	30	8	Pumpkin seed roasting	Seed Roasting	3
12	180	30	Fall photography class	Photography	4

13	75	18	Pumpkin-themed story hour	Story Hour	5
14	50	10	Scarecrow making workshop	Scarecrow Workshop	2
15	60	15	Fall festival preparation	Festival Prep	3
16	45	5	Pumpkin trivia contest	Trivia Contest	1
17	120	22	Farm-to-table cooking demo	Cooking Demo	4
18	30	6	DIY pumpkin spice candles	Candle Making	5
19	10	20	a new activity	new activity	1

Kids Activity Details:

Activityid	Guardianrequirement
1	1
3	1
5	1
6	0
7	2
8	1
9	0
13	0
14	0
18	1
19	0

Hide Results

2. DELETE Operation

Our delete operation query is demonstrated by allowing patch owners to delete an existing pumpkin patch from the PumpkinPatch table in our database.

The main query is shown below. The implementation of this query can be found in line 27-43 of the file located at /pages/delete_records.php.

```
DELETE FROM PumpkinPatch WHERE PatchID = :PatchID;
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

Our GUI doesn't list the existing PumpkinPatch table. However, you can see from screenshots in the previous section that our initial database contains 5 pumpkin patches.



Delete Pumpkin Patch

Patch ID:

1

Delete Patch

www.students.cs.ubc.ca says

Are you sure you want to delete this pumpkin patch? It will delete all data associated with the patch.

OK

Cancel

AFTER:

PumpkinPatch with ID 1 is removed from the PumpkinPatch table. As per our intended cascading effects, its associated details are also removed from the PatchMap, MapRegion, PatchTracksVariety, EquipmentLog, and MarketingPlan tables.



Delete Pumpkin Patch

Patch ID:

Delete Patch

Successfully deleted Pumpkin Patch with the given ID.
Pumpkinpatch Details:

Patchid	Patchownership	Patchsize	Patchaddress	Patchname
2	Owner2	200	Address2	Happy Farm
3	Owner3	150	Address3	Green Farm
4	Owner4	300	Address4	Organic Farm
5	Owner5	250	Address5	Country Farm

Patchmap Details:

Mapid	Patchid	Paths	Mapdescription
2	2	Path2	Description2
3	3	Path3	Description3
4	4	Path4	Description4
5	5	Path5	Description5

Mapregion Details:

Regionid	Mapid	Varietyid	Regionsize
6	2	1	25
7	2	2	25
8	2	3	25
9	2	4	25
10	2	5	25
11	3	1	30
12	3	2	15

14	3	3	25
13	3	4	7
15	3	5	1
16	4	1	50
17	4	2	35
18	4	3	28
19	4	4	17
20	4	5	44
21	5	1	75
22	5	2	2
23	5	3	60
24	5	4	75
25	5	5	80

Patchtracksvariety Details:

Patchid	Varietyid
2	2
2	3
2	4
3	3
4	4
5	1
5	2
5	3
5	4
5	5
5	6

Equipmentlog Details:

Logid	Lastmaintenancedate	Equipmentcount	Patchid
2	15-MAR-23	7	2
3	01-APR-23	6	3
4	15-APR-23	8	4
5	01-MAY-23	10	5

Marketingplan Details:

Planname	Plandescription	Socialmediarecords	Advertisingrecords	Patchid
Plan21	Description21	Record21	AdRecord21	2
Plan22	Description22	Record22	AdRecord22	2
Plan31	Description31	Record31	AdRecord31	3
Plan32	Description32	Record32	AdRecord32	3
Plan33	Description33	Record33	AdRecord33	3
Plan4	Description4	Record4	AdRecord4	4
Plan5	Description5	Record5	AdRecord5	5

Hide Results

3. UPDATE Operation

Our update operation query is demonstrated by allowing patch owners to update the details of an existing pumpkin patch.

The main query is shown below. The implementation of this query can be found in line 64-82 of the file located at /pages/update_records.php.

```
UPDATE PumpkinPatch
SET PatchOwnership = :newOwnership, PatchSize = :newSize, PatchAddress
= :newAddress, PatchName = :newName
WHERE PatchID = :PatchID;
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

The user is able to select from a drop-down table of existing pumpkin patches in the database. The user is able to enter new values for each field or check the “No change” box, which will show the existing value in that field and keep it unchanged during the update operation.



Update Pumpkin Patch

Select Pumpkin Patch:

ID: 2 - Happy Farm



New Ownership:

someone else

☐ No change

New Size:

200

☒ No change

New Address:

Address2

☒ No change

New Name:

a better name

☐ No change

Update Pumpkin Patch

AFTER:

As shown below, we have updated the details of pumpkin patch with ID 2 (previously named Happy Farm) to a new owner with a new name. Note that I did not re-start the database initialization so Pumpkin Patch with ID 1 does not show up in the result table as it has been deleted in the previous query operation.

Patch updated successfully.
Pumpkinpatch Details:

Patchid	Patchownership	Patchsize	Patchaddress	Patchname
2	someone else	200	Address2	a better name
3	Owner3	150	Address3	Green Farm
4	Owner4	300	Address4	Organic Farm
5	Owner5	250	Address5	Country Farm

Hide Results

4. Selection

Our selection query is demonstrated by allowing patch owners to search for pumpkin patches by entering any number of conditions that it needs to.

The function `constructQuery` is shown below. Note that many parts of this query are dynamically constructed. The implementation can be found in line 5-31 of the file located at /pages/search_records.php.

```
function constructQuery($conditions) {
    // QUERY 4: Selection - PumpkinPatch Table
    $query = "SELECT * FROM PumpkinPatch";
    $queryConditions = [];
    $params = [];
    $errorMessage = null;
    foreach ($conditions as $index => $condition) {
        // error checking code omitted
        $field = trim($condition['field']);
        $operator = trim($condition['operator']);
        $logicalOperator = isset($condition['logical']) && $condition['logical']
=== 'OR' ? 'OR' : 'AND';
        $param = ":value" . $index;
```

```

        $params[$param] = $operator === 'LIKE' ? "%" . $condition['value'] .
        "%" : $condition['value'];
        $queryConditions[] = ($index > 0 ? " $logicalOperator " : "") . "$field
        $operator $param";
    }
    if (!empty($queryConditions) && !$errorMessage) {
        $query .= ' WHERE ' . implode(' ', $queryConditions);
    }
    return ['query' => $query, 'params' => $params, 'error' => $errorMessage];
}

```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

The user is able to specify any number of conditions using the “Add condition” button and specify whether it is an AND or OR operation by choosing from the drop-down list.

⬅

Filter Pumpkin Patches

Patch ID ▾
Equal To ▾
2
OR ▾

Patch ID ▾
Equal To ▾
3
OR ▾
Delete

Add Condition

Filter and Get Results

AFTER:

The output result is filtered for PatchID being either 2 or 3 as requested by the user input.

Patchid	Patchownership	Patchsize	Patchaddress	Patchname
2	someone else	200	Address2	a better name
3	Owner3	150	Address3	Green Farm

5. Projection

Our projection query is demonstrated by allowing guests to view any desired attributes from any existing table in our database.

The main query is shown below, where \$selectedAttributes and \$tableName are dynamically determined from user input. The implementation of this query can be found in line 26-35 of the file located at /pages/search_records_guests.php.

```

SELECT " . implode(', ', $selectedAttributes) . " FROM " . $tableName;

```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

In this example we will use the projection tool to view the LogID, LastMaintenanceDate, and EquipmentCount from the EquipmentLog table.



Pumpkin Patch Projections

EQUIPMENTLOG ▼



LOGID



LASTMAINTENANCEDATE



EQUIPMENTCOUNT



PATCHID

Show

AFTER:

The requested data from the EquipmentLog is shown. Note that the PatchID does not show up as it was not selected by the user.

Logid	Lastmaintenancedate	Equipmentcount
2	15-MAR-23	7
3	01-APR-23	6
4	15-APR-23	8
5	01-MAY-23	10

6. Join

Our join query is demonstrated by allowing guests to search for pumpkin patches that contain patch map regions larger than or equal to a given size. This is done by joining the PumpkinPatch, PatchMap, and MapRegion tables.

The main query is shown below. The implementation of this query can be found in line 11-26 of the file located at /pages/search_map_region_guests.php.

```
"SELECT pp.PatchName, pp.PatchAddress, COUNT(mr.RegionID) AS RegionCount
FROM PumpkinPatch pp
JOIN PatchMap pm ON pp.PatchID = pm.PatchID
JOIN MapRegion mr ON pm.MapID = mr.MapID
WHERE mr.RegionSize >= :sizeThreshold
GROUP BY pp.PatchName, pp.PatchAddress";
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

User searches for pumpkin patches with map region sizes greater than or equal to a certain threshold.



Find Pumpkin Patches with Specified Minimum Map Region Size

Minimum Map Region Size:

Show Patches

AFTER:

The output lists the matching Pumpkin Patches with the corresponding Patch Names, Patch Address, and a count of how many map regions in that pumpkin patch matches the user's size requirement.

Patch Name	Patch Address	Number of Matching Map Regions
Organic Farm	Address4	3
Country Farm	Address5	4
Green Farm	Address3	1

7. Aggregation with Group By

Our aggregation with group by query is demonstrated by allowing guests to see the count of special events hosted by any existing pumpkin patch. This is done by joining the PumpkinPatch, MarketingPlan, and SpecialEvent tables, grouping the results by PatchID and PatchName, and aggregating the COUNT of EventID's for the given pumpkin patch.

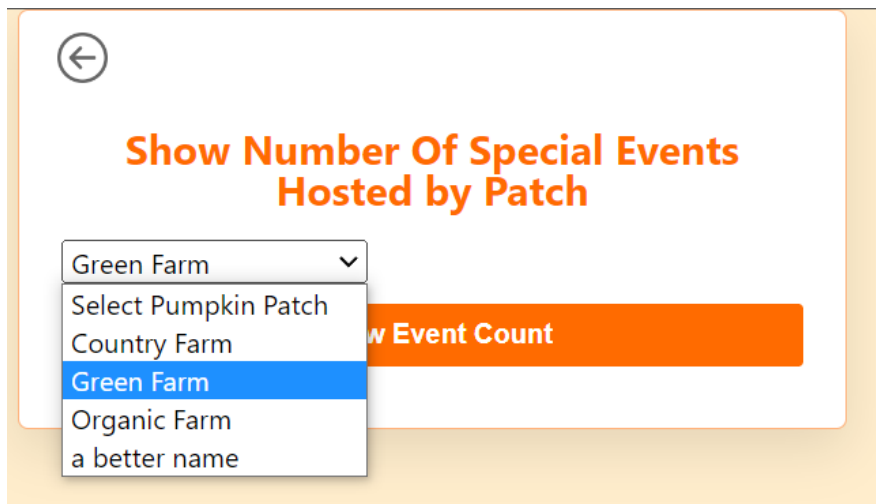
The main query is shown below. The implementation of this query can be found in line 26-41 of the file located at /pages/count_events_guests.php.

```
"SELECT pp.PatchID, pp.PatchName, COUNT(se.EventID) AS EventCount
FROM PumpkinPatch pp
JOIN MarketingPlan mp ON pp.PatchID = mp.PatchID
JOIN SpecialEvent se ON mp.PlanName = se.PlanName
WHERE pp.PatchName = :selectedPatch
GROUP BY pp.PatchID, pp.PatchName";
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

User selects from a drop-down list an existing pumpkin patch of their interest.



AFTER:

The output shows the selected patch ID, patch Name, and a count of special events hosted by the patch.

Patch ID	Patch Name	Number of Special Events Hosted
3	Green Farm	3

8. Aggregation with Having

Our aggregation with having query is demonstrated by allowing guests to search for pumpkin patches with a given minimum number of activities it features. This is done grouping the results by PatchName, aggregating the COUNT of ActivityID's for the given pumpkin patch, and filtering for results where the COUNT of ActivityID's is greater than or equal to a certain threshold.

The main query is shown below. The implementation of this query can be found in line 11-24 of the file located at /pages/search_activities_guests.php.

```
"SELECT p.PatchName, COUNT(a.ActivityID) AS NumberOfActivities
      FROM PumpkinPatch p
      JOIN Activities a ON p.PatchID = a.PatchID
      GROUP BY p.PatchName
      HAVING COUNT(a.ActivityID) >= :activityThreshold";
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

User enters a numeric value for minimum number of activities desired.



Find Pumpkin Patches with Specified Number of Activities

Minimum Number of Activities:

Find Patches

AFTER:

The output shows pumpkin patches matching the criteria along with the count of their activities. Note I have re-initialized the database to put back the previously deleted pumpkin patch in order to get a meaningful result from this query.

Patch Name	Number of Activities
Country Farm	3
Green Farm	3
Organic Farm	3
Sunny Farm	6
Happy Farm	3

9. Nested Aggregation with Group By

Our nested aggregation with group by query is demonstrated by allowing guests to see the average activity fees of each pumpkin patch sorted from least to most expensive. This is done by aggregating the rounded average fee for each patch, filtering only the patches where average fee is greater than or equal to the minimum average fee, and showing results in ascending order.

The main query is shown below. The implementation of this query can be found in line 11-29 of the file located at /pages/average_activity_fee.php.

```
"SELECT p.PatchName, ROUND(AVG(a.Fee), 2) AS avg_fee
FROM PumpkinPatch p
JOIN Activities a ON p.PatchID = a.PatchID
GROUP BY p.PatchName
HAVING AVG(a.Fee) >= (
    SELECT MIN(AVG(a2.Fee))
    FROM Activities a2
    JOIN PumpkinPatch p2 ON a2.PatchID = p2.PatchID
    GROUP BY p2.PatchID
)
ORDER BY AVG(a.Fee)";
```

See screenshots below demonstrating the use of this query.

BEFORE, DURING & AFTER:

As no user input is required, the user automatically presented with the average fee information for all existing patches currently in the database.

[View Average Activity Fees](#)



Average Pumpkin Patch Activity Fees

Pumpkin Patch Activity Fees sorted from Minimum to Maximum Average Fees

Patch Name	Sorted Average Activity Fee
Green Farm	10
Country Farm	12
Happy Farm	15
Organic Farm	22.33
Sunny Farm	25

10. Division

Our division query is demonstrated by allowing guests to find pumpkin patches that plant all types of pumpkin varieties tracked by our database. This is done by first retrieving all VarietyID's from the PumpkinVariety table, then using a subquery to retrieve all VarietyID's from the PatchTracksVariety table for the current pumpkin patch (matched by PatchID). If any pumpkin patch plants all of the tracked pumpkin varieties, then the two sets of VarietyID's in the previous operation would be the same. So, MINUS is used to take the set difference between these 2 sets of VarietyID values. If there is no difference, the NOT EXISTS condition would return true, and therefore returning the subject pumpkin patch in our filtered results.

The main query is shown below. The implementation of this query can be found in line 27-45 of the file located at /pages/search_variety_guests.php.

```
"SELECT pp.PatchID, pp.PatchName, pp.PatchAddress
FROM PumpkinPatch pp
WHERE NOT EXISTS (
    (
        SELECT VarietyID
        FROM PumpkinVariety
    ) MINUS (
        SELECT ptv.VarietyID
        FROM PatchTracksVariety ptv
        WHERE pp.PatchID = ptv.PatchID
    )
)";
```

See screenshots below demonstrating the use of this query.

BEFORE & DURING:

The user is presented with a list of available varieties currently tracked by our database.



Pumpkin Patches with All Varieties Planted

Available Varieties:

- Big Max
- Cinderella
- Jarrahdale
- Kabocha
- Something Else
- Sugar Pie

Search

AFTER:

The output shows the user information on the pumpkin patches that plants all of the currently tracked varieties. If our database is modified such that varieties are added/deleted in the PumpkinVarieties table or in the PatchTracksVariety table, the output would reflect these changes.

Patch ID	Patch Name	Patch Address
1	Sunny Farm	Address1
5	Country Farm	Address5

Appendices

Appendix A - SQL Script

```
-- Disable foreign key checks to avoid errors during dropping
BEGIN
    EXECUTE IMMEDIATE 'ALTER SESSION SET "_ORACLE_SCRIPT"=true';
    EXECUTE IMMEDIATE 'ALTER SESSION SET CONSTRAINTS = DISABLED';
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.put_line('Failed to disable constraints: ' || SQLERRM);
END;
/

-- Drop tables and sequences if they exist
DECLARE
    table_not_exist EXCEPTION;
    PRAGMA EXCEPTION_INIT(table_not_exist, -942);
    sequence_not_exist EXCEPTION;
    PRAGMA EXCEPTION_INIT(sequence_not_exist, -2289);
BEGIN
    -- Drop tables in the reverse order of dependencies
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE PumpkinPatch CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE PatchMap CASCADE CONSTRAINTS'; EXCEPTION
WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE PumpkinVariety CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE MapRegion CASCADE CONSTRAINTS'; EXCEPTION
WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE HarvestSchedule CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE PatchTracksVariety CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE EquipmentLog CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE MaintenanceSchedule CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE MarketingPlan CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE SpecialEvent CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE Tickets CASCADE CONSTRAINTS'; EXCEPTION
WHEN table_not_exist THEN NULL; END;
```

```

    BEGIN EXECUTE IMMEDIATE 'DROP TABLE Activities CASCADE CONSTRAINTS'; EXCEPTION
WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE KidsActivities CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    BEGIN EXECUTE IMMEDIATE 'DROP TABLE AdultActivities CASCADE CONSTRAINTS';
EXCEPTION WHEN table_not_exist THEN NULL; END;
    -- Drop the sequence
    BEGIN EXECUTE IMMEDIATE 'DROP SEQUENCE ACTIVITIES_SEQ'; EXCEPTION WHEN
sequence_not_exist THEN NULL; END;
END;
/

-- Create Sequence for auto-incrementing IDs
CREATE SEQUENCE ACTIVITIES_SEQ
    MINVALUE 1
    START WITH 19
    INCREMENT BY 1
/

-- create tables
CREATE TABLE PumpkinPatch (
    PatchID INTEGER PRIMARY KEY,
    PatchOwnership VARCHAR2(25),
    PatchSize INTEGER,
    PatchAddress VARCHAR2(25),
    PatchName VARCHAR2(25) NOT NULL UNIQUE
);

CREATE TABLE PumpkinVariety (
    VarietyID INTEGER PRIMARY KEY,
    QuantityPlanted INTEGER,
    PlantedDate DATE,
    VarietyName VARCHAR2(25) UNIQUE NOT NULL
);

-- added UNIQUE constraint to PatchID due to one-to-one relationship
CREATE TABLE PatchMap (
    MapID INTEGER PRIMARY KEY,
    PatchID INTEGER NOT NULL UNIQUE,
    Paths VARCHAR2(25),
    MapDescription VARCHAR2(25),
    FOREIGN KEY (PatchID) REFERENCES PumpkinPatch(PatchID) ON DELETE CASCADE
);

-- updated primary key to include both MapID and RegionID as it is a weak entity

```

```

CREATE TABLE MapRegion (
    RegionID INTEGER NOT NULL,
    MapID INTEGER NOT NULL,
    VarietyID INTEGER,
    RegionSize INTEGER,
    PRIMARY KEY (RegionID, MapID),
    FOREIGN KEY (MapID) REFERENCES PatchMap(MapID) ON DELETE CASCADE,
    FOREIGN KEY (VarietyID) REFERENCES PumpkinVariety(VarietyID)
);

CREATE TABLE HarvestSchedule (
    PlantedDate DATE,
    VarietyName VARCHAR2(25),
    HarvestDate DATE,
    PRIMARY KEY (PlantedDate, VarietyName)
);

CREATE TABLE PatchTracksVariety (
    PatchID INTEGER,
    VarietyID INTEGER,
    PRIMARY KEY (PatchID, VarietyID),
    FOREIGN KEY (PatchID) REFERENCES PumpkinPatch(PatchID) ON DELETE CASCADE,
    FOREIGN KEY (VarietyID) REFERENCES PumpkinVariety(VarietyID) ON DELETE
CASCADE
);

CREATE TABLE MaintenanceSchedule (
    LastMaintenanceDate DATE PRIMARY KEY,
    NextMaintenanceDate DATE
);

-- added UNIQUE constraint to PatchID due to one-to-one relationship
CREATE TABLE EquipmentLog (
    LogID INTEGER PRIMARY KEY,
    LastMaintenanceDate DATE,
    EquipmentCount INTEGER,
    PatchID INTEGER UNIQUE,
    FOREIGN KEY (PatchID) REFERENCES PumpkinPatch(PatchID) ON DELETE CASCADE
);

CREATE TABLE MarketingPlan (
    PlanName VARCHAR2(25) PRIMARY KEY,
    PlanDescription VARCHAR2(25),
    SocialMediaRecords VARCHAR2(25),
    AdvertisingRecords VARCHAR2(25),

```



```

    PatchID INTEGER,
    FOREIGN KEY (PatchID) REFERENCES PumpkinPatch(PatchID) ON DELETE CASCADE
);

CREATE TABLE SpecialEvent (
    EventID INTEGER PRIMARY KEY,
    EventName VARCHAR2(30) NOT NULL,
    PlanName VARCHAR2(30) NOT NULL,
    FOREIGN KEY (PlanName) REFERENCES MarketingPlan(PlanName) ON DELETE CASCADE
);

-- updated primary key to include both TicketID and EventID as it is a weak
entity
CREATE TABLE Tickets (
    TicketID INTEGER NOT NULL,
    EventID INTEGER NOT NULL,
    AdmissionType VARCHAR2(25),
    VisitorType VARCHAR2(25),
    PRIMARY KEY (TicketID, EventID),
    FOREIGN KEY (EventID) REFERENCES SpecialEvent(EventID) ON DELETE CASCADE
);

CREATE TABLE Activities (
    ActivityID INTEGER PRIMARY KEY,
    Duration INTEGER,
    Fee INTEGER,
    ActivityDescription VARCHAR2(30),
    ActivityName VARCHAR2(25) NOT NULL,
    PatchID INTEGER,
    FOREIGN KEY (PatchID) REFERENCES PumpkinPatch(PatchID) ON DELETE CASCADE
);

CREATE TABLE KidsActivities (
    ActivityID INTEGER NOT NULL,
    GuardianRequirement INTEGER,
    PRIMARY KEY (ActivityID),
    FOREIGN KEY (ActivityID) REFERENCES Activities(ActivityID) ON DELETE CASCADE
);

CREATE TABLE AdultActivities (
    ActivityID INTEGER NOT NULL,
    AgeRequirement INTEGER,
    AlcoholInvolvement INTEGER,
    PRIMARY KEY (ActivityID),
    FOREIGN KEY (ActivityID) REFERENCES Activities(ActivityID) ON DELETE CASCADE
);

```

```

);

-- populate tables
-- Inserting into PumpkinPatch table
INSERT ALL
  INTO PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)
VALUES (1, 'Owner1', 100, 'Address1', 'Sunny Farm')
  INTO PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)
VALUES (2, 'Owner2', 200, 'Address2', 'Happy Farm')
  INTO PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)
VALUES (3, 'Owner3', 150, 'Address3', 'Green Farm')
  INTO PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)
VALUES (4, 'Owner4', 300, 'Address4', 'Organic Farm')
  INTO PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)
VALUES (5, 'Owner5', 250, 'Address5', 'Country Farm')
SELECT * FROM dual;

-- Inserting into PumpkinVariety table
INSERT ALL
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (1, 50, TO_DATE('2023-04-01', 'YYYY-MM-DD'), 'Big Max')
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (2, 75, TO_DATE('2023-04-15', 'YYYY-MM-DD'), 'Sugar Pie')
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (3, 65, TO_DATE('2023-04-20', 'YYYY-MM-DD'), 'Cinderella')
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (4, 80, TO_DATE('2023-05-01', 'YYYY-MM-DD'), 'Jarrahdale')
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (5, 100, TO_DATE('2023-05-10', 'YYYY-MM-DD'), 'Kabocha')
  INTO PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)
VALUES (6, 100, TO_DATE('2023-05-10', 'YYYY-MM-DD'), 'Something Else')
SELECT * FROM dual;

-- Inserting into PatchMap table
INSERT ALL
  INTO PatchMap (MapID, PatchID, Paths, MapDescription) VALUES (1, 1, 'Path1',
'Description1')
  INTO PatchMap (MapID, PatchID, Paths, MapDescription) VALUES (2, 2, 'Path2',
'Description2')
  INTO PatchMap (MapID, PatchID, Paths, MapDescription) VALUES (3, 3, 'Path3',
'Description3')
  INTO PatchMap (MapID, PatchID, Paths, MapDescription) VALUES (4, 4, 'Path4',
'Description4')
  INTO PatchMap (MapID, PatchID, Paths, MapDescription) VALUES (5, 5, 'Path5',
'Description5')

```

```

SELECT * FROM dual;

-- Inserting into MapRegion table
INSERT ALL
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (1, 1, 1, 20)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (2, 1, 2, 10)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (3, 1, 3, 5)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (4, 1, 4, 2)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (5, 1, 5, 18)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (6, 2, 1, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (7, 2, 2, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (8, 2, 3, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (9, 2, 4, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (10, 2, 5, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (11, 3, 1, 30)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (12, 3, 2, 15)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (14, 3, 3, 25)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (13, 3, 4, 7)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (15, 3, 5, 1)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (16, 4, 1, 50)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (17, 4, 2, 35)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (18, 4, 3, 28)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (19, 4, 4, 17)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (20, 4, 5, 44)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (21, 5, 1, 75)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (22, 5, 2, 2)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (23, 5, 3, 60)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (24, 5, 4, 75)
  INTO MapRegion (RegionID, MapID, VarietyID, RegionSize) VALUES (25, 5, 5, 80)
SELECT * FROM dual;

-- Inserting into HarvestSchedule table
INSERT ALL
  INTO HarvestSchedule (PlantedDate, VarietyName, HarvestDate) VALUES
  (TO_DATE('2023-04-01', 'YYYY-MM-DD'), 'Big Max', TO_DATE('2023-10-01', 'YYYY-MM-DD'))
  INTO HarvestSchedule (PlantedDate, VarietyName, HarvestDate) VALUES
  (TO_DATE('2023-04-15', 'YYYY-MM-DD'), 'Sugar Pie', TO_DATE('2023-10-15', 'YYYY-MM-DD'))
  INTO HarvestSchedule (PlantedDate, VarietyName, HarvestDate) VALUES
  (TO_DATE('2023-04-20', 'YYYY-MM-DD'), 'Cinderella', TO_DATE('2023-10-20', 'YYYY-MM-DD'))
  INTO HarvestSchedule (PlantedDate, VarietyName, HarvestDate) VALUES
  (TO_DATE('2023-05-01', 'YYYY-MM-DD'), 'Jarrahdale', TO_DATE('2023-11-01', 'YYYY-MM-DD'))

```

```

    INTO HarvestSchedule (PlantedDate, VarietyName, HarvestDate) VALUES
(TO_DATE('2023-05-10', 'YYYY-MM-DD'), 'Kabocha', TO_DATE('2023-11-10', 'YYYY-MM-
DD'))
SELECT * FROM dual;

-- Inserting into PatchTracksVariety table
INSERT ALL
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 1)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 2)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 3)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 4)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 5)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (1, 6)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (2, 2)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (2, 3)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (2, 4)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (3, 3)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (4, 4)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 1)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 2)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 3)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 4)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 5)
    INTO PatchTracksVariety (PatchID, VarietyID) VALUES (5, 6)
SELECT * FROM dual;

-- Inserting into MaintenanceSchedule table
INSERT ALL
    INTO MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate) VALUES
(TO_DATE('2023-03-01', 'YYYY-MM-DD'), TO_DATE('2023-09-01', 'YYYY-MM-DD'))
    INTO MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate) VALUES
(TO_DATE('2023-03-15', 'YYYY-MM-DD'), TO_DATE('2023-09-15', 'YYYY-MM-DD'))
    INTO MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate) VALUES
(TO_DATE('2023-04-01', 'YYYY-MM-DD'), TO_DATE('2023-10-01', 'YYYY-MM-DD'))
    INTO MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate) VALUES
(TO_DATE('2023-04-15', 'YYYY-MM-DD'), TO_DATE('2023-10-15', 'YYYY-MM-DD'))
    INTO MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate) VALUES
(TO_DATE('2023-05-01', 'YYYY-MM-DD'), TO_DATE('2023-11-01', 'YYYY-MM-DD'))
SELECT * FROM dual;

-- Inserting into EquipmentLog table
INSERT ALL
    INTO EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, PatchID) VALUES
(1, TO_DATE('2023-03-01', 'YYYY-MM-DD'), 5, 1)

```

```

    INTO EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, PatchID) VALUES
(2, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 7, 2)
    INTO EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, PatchID) VALUES
(3, TO_DATE('2023-04-01', 'YYYY-MM-DD'), 6, 3)
    INTO EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, PatchID) VALUES
(4, TO_DATE('2023-04-15', 'YYYY-MM-DD'), 8, 4)
    INTO EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, PatchID) VALUES
(5, TO_DATE('2023-05-01', 'YYYY-MM-DD'), 10, 5)
SELECT * FROM dual;

-- Inserting into MarketingPlan table
INSERT ALL
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan1', 'Description1', 'Record1',
'AdRecord1', 1)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan21', 'Description21', 'Record21',
'AdRecord21', 2)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan22', 'Description22', 'Record22',
'AdRecord22', 2)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan31', 'Description31', 'Record31',
'AdRecord31', 3)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan32', 'Description32', 'Record32',
'AdRecord32', 3)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan33', 'Description33', 'Record33',
'AdRecord33', 3)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan4', 'Description4', 'Record4',
'AdRecord4', 4)
    INTO MarketingPlan (PlanName, PlanDescription, SocialMediaRecords,
AdvertisingRecords, PatchID) VALUES ('Plan5', 'Description5', 'Record5',
'AdRecord5', 5)
SELECT * FROM dual;

-- Inserting into SpecialEvent table
INSERT ALL
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (1, 'Halloween Bash',
'Plan1')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (211, 'Harvest
Festival1,T', 'Plan21')

```

```

    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (212, 'Harvest
Festival2', 'Plan21')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (221, 'Harvest
Festival3', 'Plan22')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (222, 'Harvest
Festival4', 'Plan22')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (31, 'Pumpkin Carving
Contest1', 'Plan31')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (32, 'Pumpkin Carving
Contest2', 'Plan32')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (33, 'Pumpkin Carving
Contest3', 'Plan33')
    INTO SpecialEvent (EventID, EventName, PlanName) VALUES (4, 'Thanksgiving
Sale', 'Plan4')
SELECT * FROM dual;

-- Inserting into Tickets table
INSERT ALL
    INTO Tickets (TicketID, EventID, AdmissionType, VisitorType) VALUES (1, 1,
'Standard', 'Adult')
    INTO Tickets (TicketID, EventID, AdmissionType, VisitorType) VALUES (2, 211,
'Premium', 'Child')
    INTO Tickets (TicketID, EventID, AdmissionType, VisitorType) VALUES (3, 31,
'Standard', 'Senior')
    INTO Tickets (TicketID, EventID, AdmissionType, VisitorType) VALUES (4, 4,
'Standard', 'Adult')
    INTO Tickets (TicketID, EventID, AdmissionType, VisitorType) VALUES (5, 221,
'Premium', 'Adult')
SELECT * FROM dual;

-- Inserting into Activities table
INSERT ALL
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (1, 30, 5, 'Pumpkin carving activity', 'Carving Class', 1)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (2, 60, 10, 'Pumpkin pie making activity', 'Cooking Class', 2)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (3, 45, 7, 'Hayrides around the patch', 'Hayride', 3)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (4, 120, 15, 'Halloween movie night', 'Movie Night', 4)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (5, 90, 12, 'Pumpkin painting activity', 'Painting Class', 5)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (6, 90, 20, 'Youth art class', 'Youth Art Class', 1)

```

```

    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (7, 120, 50, 'Wine tasting event', 'Winery', 1)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (8, 60, 10, 'Cultural dance showcase', 'Dance Fiesta', 1)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (9, 150, 60, 'Cooking class for adults', 'Culinary Masters', 1)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (10, 120, 25, 'Nighttime nature walk', 'Nature Walk', 2)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (11, 30, 8, 'Pumpkin seed roasting', 'Seed Roasting', 3)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (12, 180, 30, 'Fall photography class', 'Photography', 4)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (13, 75, 18, 'Pumpkin-themed story hour', 'Story Hour', 5)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (14, 50, 10, 'Scarecrow making workshop', 'Scarecrow Workshop',
2)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (15, 60, 15, 'Fall festival preparation', 'Festival Prep', 3)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (16, 45, 5, 'Pumpkin trivia contest', 'Trivia Contest', 1)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (17, 120, 22, 'Farm-to-table cooking demo', 'Cooking Demo', 4)
    INTO Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName,
PatchID) VALUES (18, 30, 6, 'DIY pumpkin spice candles', 'Candle Making', 5)
SELECT * FROM dual;

```

```
-- Inserting into KidsActivities table
```

```
INSERT ALL
```

```

    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (1, 1)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (3, 1)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (5, 1)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (6, 0)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (7, 2)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (8, 1)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (9, 0)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (13, 0)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (14, 0)
    INTO KidsActivities (ActivityID, GuardianRequirement) VALUES (18, 1)

```

```
SELECT * FROM dual;
```

```
-- Inserting into AdultActivities table
```

```
INSERT ALL
```

```

    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(2, 21, 0)

```

```

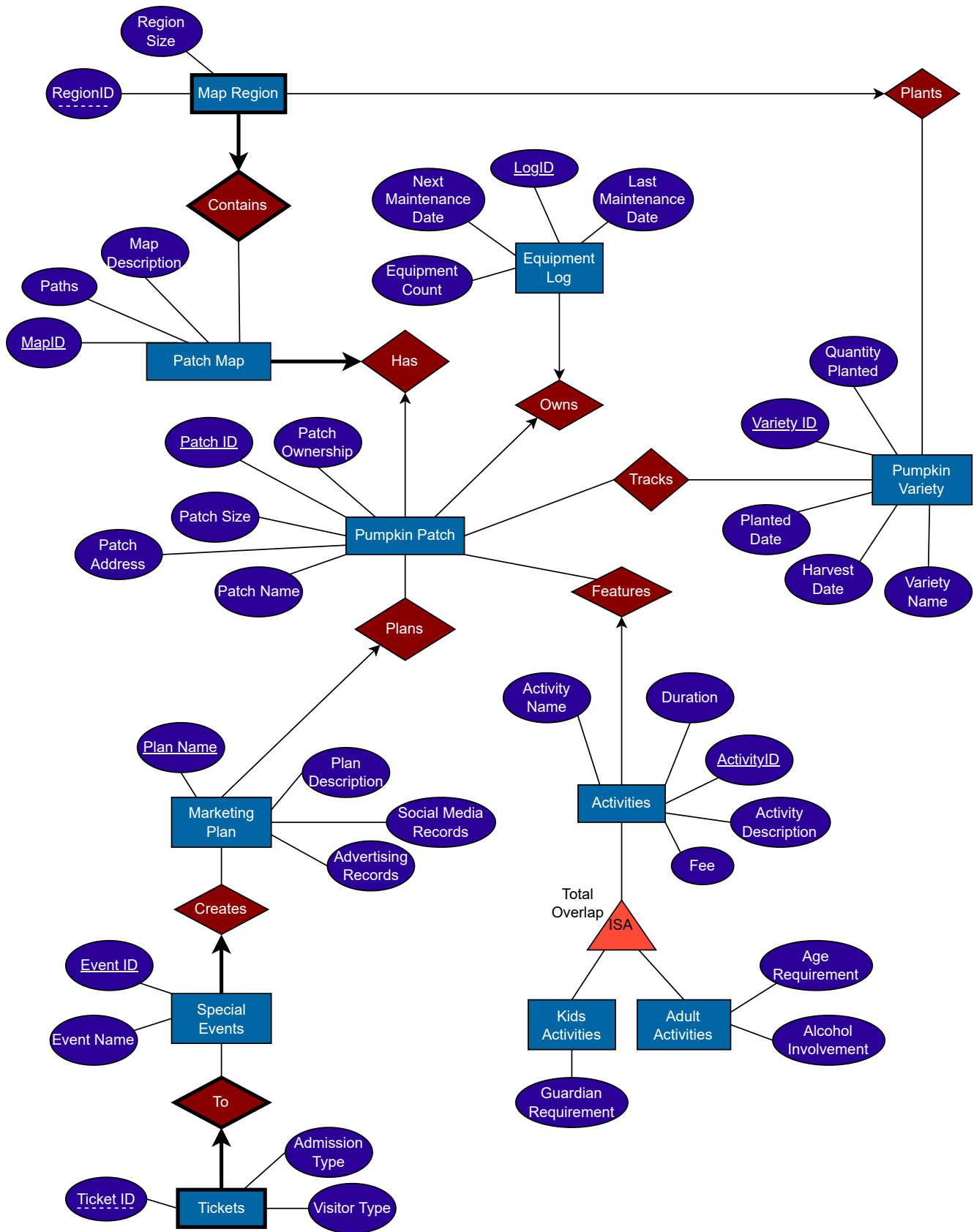
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(4, 21, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(6, 0, 0)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(7, 0, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(8, 0, 0)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(9, 0, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(10, 18, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(11, 18, 0)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(12, 20, 0)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(14, 0, 0)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(15, 20, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(16, 22, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(17, 23, 1)
    INTO AdultActivities (ActivityID, AgeRequirement, AlcoholInvolvement) VALUES
(18, 0, 0)
SELECT * FROM dual;

-- Enable constraints back
BEGIN
    EXECUTE IMMEDIATE 'ALTER SESSION SET CONSTRAINTS = ENABLED';
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.put_line('Failed to enable constraints: ' || SQLERRM);
END;
/

COMMIT;
/

```


Appendix B - ER Diagram



Appendix C - Schema Screenshots from SQL Plus

- PumpkinPatch (PatchID, PatchOwnership, PatchSize, PatchAddress, PatchName)

```
SQL> SELECT * FROM PumpkinPatch;
```

PATCHID	PATCHOWNERSHIP	PATCHSIZE	PATCHADDRESS
1	Owner1	100	Address1
Sunny Farm			
2	Owner2	200	Address2
Happy Farm			
3	Owner3	150	Address3
Green Farm			
4	Owner4	300	Address4
Organic Farm			
5	Owner5	250	Address5
Country Farm			

```
SQL>
```

- PumpkinVariety (VarietyID, QuantityPlanted, PlantedDate, VarietyName)

```
SQL> SELECT * FROM PumpkinVariety;
```

VARIETYID	QUANTITYPLANTED	PLANTEDDA	VARIETYNAME
1	50	01-APR-23	Big Max
2	75	15-APR-23	Sugar Pie
3	65	20-APR-23	Cinderella
4	80	01-MAY-23	Jarrahdale
5	100	10-MAY-23	Kabocha
6	100	10-MAY-23	Something Else

6 rows selected.

```
SQL>
```

- PatchMap (MapID, **PatchID**, Paths, MapDescription)

```
SQL> SELECT * FROM PatchMap;
```

MAPID	PATCHID	PATHS	MAPDESCRIPTION
1	1	Path1	Description1
2	2	Path2	Description2
3	3	Path3	Description3
4	4	Path4	Description4
5	5	Path5	Description5

```
SQL> 
```

- MapRegion (RegionID, MapID, VarietyID, RegionSize)

```
SQL> SELECT * FROM MapRegion;
```

REGIONID	MAPID	VARIETYID	REGIONSIZE
1	1	1	20
2	1	2	10
3	1	3	5
4	1	4	2
5	1	5	18
6	2	1	25
7	2	2	25
8	2	3	25
9	2	4	25
10	2	5	25
11	3	1	30
12	3	2	15
14	3	3	25
13	3	4	7
15	3	5	1
16	4	1	50
17	4	2	35
18	4	3	28
19	4	4	17
20	4	5	44
21	5	1	75
22	5	2	2
23	5	3	60
24	5	4	75
25	5	5	80

25 rows selected.

```
SQL> 
```

- HarvestSchedule (PlantedDate, VarietyName, HarvestDate)

```
SQL> SELECT * FROM HarvestSchedule;
```

PLANTEDDA	VARIETYNAME	HARVESTDA
01-APR-23	Big Max	01-OCT-23
15-APR-23	Sugar Pie	15-OCT-23
20-APR-23	Cinderella	20-OCT-23
01-MAY-23	Jarrahdale	01-NOV-23
10-MAY-23	Kabocha	10-NOV-23

```
SQL> 
```

- PatchTracksVariety (PatchID, VarietyID)

```
SQL> SELECT * FROM PatchTracksVariety;
```

PATCHID	VARIETYID
1	1
1	2
1	3
1	4
1	5
1	6
2	2
2	3
2	4
3	3
4	4

PATCHID	VARIETYID
5	1
5	2
5	3
5	4
5	5
5	6

17 rows selected.

```
SQL> 
```

- MaintenanceSchedule (LastMaintenanceDate, NextMaintenanceDate)

```
SQL> SELECT * FROM MaintenanceSchedule;

LASTMAINT  NEXTMAINT
-----
01-MAR-23  01-SEP-23
15-MAR-23  15-SEP-23
01-APR-23  01-OCT-23
15-APR-23  15-OCT-23
01-MAY-23  01-NOV-23

SQL> █
```

- EquipmentLog (LogID, LastMaintenanceDate, EquipmentCount, **PatchID**)

```
SQL> SELECT * FROM EquipmentLog;

LOGID  LASTMAINT  EQUIPMENTCOUNT  PATCHID
-----
1 01-MAR-23          5           1
2 15-MAR-23          7           2
3 01-APR-23          6           3
4 15-APR-23          8           4
5 01-MAY-23         10           5

SQL> █
```

- MarketingPlan (PlanName, PlanDescription, SocialMediaRecords, AdvertisingRecords, PatchID)

```
SQL> SELECT * FROM MarketingPlan;
```

PLANNAME	PLANDescription	SOCIALMEDIARECORDS
ADVERTISINGRECORDS	PATCHID	
Plan1	Description1	Record1
AdRecord1	1	
Plan21	Description21	Record21
AdRecord21	2	
Plan22	Description22	Record22
AdRecord22	2	
Plan31	Description31	Record31
AdRecord31	3	
Plan32	Description32	Record32
AdRecord32	3	
Plan33	Description33	Record33
AdRecord33	3	
Plan4	Description4	Record4
AdRecord4	4	
Plan5	Description5	Record5
AdRecord5	5	

8 rows selected.

```
SQL> 
```

- SpecialEvent (EventID, EventName, PlanName)

```
SQL> SELECT * FROM SpecialEvent;
```

EVENTID	EVENTNAME	PLANNAME
1	Halloween Bash	Plan1
211	Harvest Festival1,T	Plan21
212	Harvest Festival2	Plan21
221	Harvest Festival3	Plan22
222	Harvest Festival4	Plan22
31	Pumpkin Carving Contest1	Plan31
32	Pumpkin Carving Contest2	Plan32
33	Pumpkin Carving Contest3	Plan33
4	Thanksgiving Sale	Plan4

9 rows selected.

```
SQL> █
```

- Tickets (TicketID, EventID, AdmissionType, VisitorType)

```
SQL> SELECT * FROM Tickets;
```

TICKETID	EVENTID	ADMISSIONTYPE	VISITORTYPE
1	1	Standard	Adult
2	211	Premium	Child
3	31	Standard	Senior
4	4	Standard	Adult
5	221	Premium	Adult

```
SQL> █
```

- Activities (ActivityID, Duration, Fee, ActivityDescription, ActivityName, **PatchID**)

```
SQL> SELECT * FROM Activities;
```

ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION

ACTIVITYNAME		PATCHID	

1	30	5	Pumpkin carving activity
Carving Class		1	

2	60	10	Pumpkin pie making activity
Cooking Class		2	

3	45	7	Hayrides around the patch
Hayride		3	

ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION

ACTIVITYNAME		PATCHID	

4	120	15	Halloween movie night
Movie Night		4	

5	90	12	Pumpkin painting activity
Painting Class		5	

6	90	20	Youth art class
Youth Art Class		1	

ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION

ACTIVITYNAME		PATCHID	

7	120	50	Wine tasting event
Winery		1	

8	60	10	Cultural dance showcase
Dance Fiesta		1	

9	150	60	Cooking class for adults
Culinary Masters		1	

ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION
ACTIVITYNAME		PATCHID	
10	120	25	Nighttime nature walk
Nature Walk		2	
11	30	8	Pumpkin seed roasting
Seed Roasting		3	
12	180	30	Fall photography class
Photography		4	
ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION
ACTIVITYNAME		PATCHID	
13	75	18	Pumpkin-themed story hour
Story Hour		5	
14	50	10	Scarecrow making workshop
Scarecrow Workshop		2	
15	60	15	Fall festival preparation
Festival Prep		3	
ACTIVITYID	DURATION	FEE	ACTIVITYDESCRIPTION
ACTIVITYNAME		PATCHID	
16	45	5	Pumpkin trivia contest
Trivia Contest		1	
17	120	22	Farm-to-table cooking demo
Cooking Demo		4	
18	30	6	DIY pumpkin spice candles
Candle Making		5	

18 rows selected.

SQL> █

- KidsActivities (ActivityID, Guardian Requirement)

```
SQL> SELECT * FROM KidsActivities;

ACTIVITYID  GUARDIANREQUIREMENT
-----
          1              1
          3              1
          5              1
          6              0
          7              2
          8              1
          9              0
         13              0
         14              0
         18              1

10 rows selected.

SQL> 
```

- AdultActivities (ActivityID, Age Requirement, Alcohol Involvement)

```
SQL> SELECT * FROM AdultActivities;

ACTIVITYID  AGEREQUIREMENT  ALCOHOLINVOLVEMENT
-----
          2             21              0
          4             21              1
          6             0              0
          7             0              1
          8             0              0
          9             0              1
         10            18              1
         11            18              0
         12            20              0
         14             0              0
         15            20              1

ACTIVITYID  AGEREQUIREMENT  ALCOHOLINVOLVEMENT
-----
         16            22              1
         17            23              1
         18             0              0

14 rows selected.

SQL> 
```