Project Name: Amazon’s Alexa- A voice controlled Virtual Assistant Schema

BUAN 6320.501 | Group 9 | Fall 2024

Deepika Boppana | Thriksha Giriraju | Masood Hyder | Laetitia Le | Yash Thakkar

Project Overview:

- **Objective:**

The objective of Amazon’s Alexa is to provide users with a seamless, hands-free experience for accessing information, managing daily tasks, controlling smart home devices, and enjoying entertainment through voice interaction. By utilizing advanced natural language processing and machine learning, Alexa aims to understand and respond accurately to user queries and commands, improving personalization over time. It also serves as an extensible platform for developers, allowing for the creation of custom skills to expand Alexa's capabilities. Alexa prioritizes user privacy and security, ensuring a safe and intuitive digital assistant experience across compatible devices.

- **Scope:**

The scope of Amazon’s Alexa voice-controlled virtual assistant includes enabling voice interaction for tasks such as managing smart home devices, retrieving information, facilitating shopping, and providing personalized experiences through third-party integrations and media control. Alexa will support natural language processing, continuous learning for improved interactions, and robust privacy and security measures. It will also offer a platform for developers to expand functionality with custom skills, ensuring adaptability to evolving user needs.

Business Requirements:

- A user can issue multiplevoice commands.
- A user can have a primary device assigned
- A user can own multiple devices
- A user can enable multiple skills, and a skil can be used by multiple users.
- A voice command can lead to multiple interactions
- A skill can be invoked in multiple interactions.

1. Entity and Attribute Description (15 Points)

Each entity in this database schema represents a core component of Amazon Alexa's interactions, carefully selected to capture meaningful user, device, command, and skill data. The entity choices align with Alexa's operational workflow, from users issuing commands to interactions being recorded and skills being utilized.

Entity: User

- **Attributes:**
  - **UserID (Primary Key)**: Unique identifier for each user, ensuring distinctness across records.
  - **Name** and **Email**: Basic identifying details. Email also serves as a unique form of contact.
  - **RegistrationDate**: Marks when the user registered with Alexa, useful for analyzing user engagement over time.
  - **DeviceType** and **Location**: Optional attributes that provide further context for user interactions. DeviceType captures the main type of Alexa device registered, while Location provides geographical insights.
- **Usage:** This entity forms the backbone of the database by uniquely identifying each user, allowing for personalized insights and analytics. It's used across various relationships, connecting to VoiceCommand, Device, and Skill entities to track and analyze Alexa usage patterns.

Entity: VoiceCommand

- **Attributes:**
  - **CommandID (Primary Key)**: Unique identifier for each voice command, ensuring accurate tracking.
  - **CommandText**: The content of the command, such as "Play music" or "What's the weather?", crucial for understanding user intent.
  - **Timestamp**: Records the exact time of the command, allowing for sequential analysis.
  - **ResponseTime**, **SuccessStatus**, and **VoiceProfile**: Optional but valuable attributes to analyze command effectiveness, user satisfaction, and personalization through voice profiles.
- **Usage:** VoiceCommand allows for capturing user interaction specifics with Alexa. It's linked to Interaction to provide insights into each unique command, which is essential for performance monitoring and quality assessment.

Entity: Skill

- **Attributes:**
  - **SkillID (Primary Key)** and **SkillName**: Identify each skill within Alexa, such as "Weather" or "Music".
  - **SkillCategory**, **LaunchCount**, **UserRating**, and **Developer**: Optional attributes that detail skill type, usage frequency, user satisfaction, and developer information. These attributes enable skill analytics.
- **Usage:** Skills represent functionalities that users interact with. By linking Skills to Interaction and User, we can monitor usage patterns, user engagement with specific skills, and the performance of different skill categories.

Entity: Device

- **Attributes:**
  - **DeviceID (Primary Key)**: Unique identifier for each Alexa device.
  - **DeviceName** and **DeviceModel**: Device characteristics, providing insights into device popularity and performance.
  - **UserID (Foreign Key)**: Connects each device to a user, enabling a direct link to user-related interactions.
  - **LastUsed** and **PurchaseDate**: Optional timestamps that allow for the tracking of device usage frequency and lifecycle.
- **Usage:** Device captures details about Alexa hardware used by each user. Tracking device usage through attributes like LastUsed offers insights into device reliability and user interaction frequency.

Entity: Interaction

- **Attributes:**
  - **InteractionID (Primary Key)**: Unique identifier for each interaction session, encapsulating a specific user command or skill.
  - **UserID (Foreign Key)**, **CommandID (Foreign Key)**, **SkillID (Foreign Key)**: Link each interaction to specific users, commands, and skills, creating a comprehensive record.
  - **Duration** and **InteractionDate**: Optional metrics that help analyze engagement and the time spent in each interaction.
- **Usage:** Interaction links users with commands and skills, enabling complex analysis on how users engage with Alexa's capabilities and for how long.

Entity: UserSkill (Junction Table)

- **Attributes:**
  - **UserID (Foreign Key)**, **SkillID (Foreign Key)**: Links users and skills for a many-to-many relationship.
  - **DateEnabled**: Tracks when the user enabled each skill.
- **Usage:** UserSkill records which users have enabled which skills, essential for tracking personalized skill usage.

Entity: DeviceCommand (Junction Table)

- **Attributes:**
  - **DeviceID (Foreign Key)**, **CommandID (Foreign Key)**: Tracks the relationship between devices and commands.
  - **UsageCount**: Counts how often a command was issued from a specific device.
- **Usage:** DeviceCommand records command executions across different devices, aiding in device-specific command analysis.

Entity: SkillUsage (Junction Table)

- **Attributes:**
  - **SkillID (Foreign Key)**, **InteractionID (Foreign Key)**: Tracks which interactions involved which skills.
  - **UsageFrequency**: Indicates how frequently the skill was used.
- **Usage:** SkillUsage captures the depth of skill engagement across interactions, supporting skill performance evaluation.

2. Relationship and Cardinality Description (15 Points)

This database design utilizes relationships to capture the ways in which users, devices, commands, and skills interact.

- **User to VoiceCommand**:
  - **Relationship**: One-to-Many
  - **Description**: Each user can issue multiple voice commands, but each command belongs to only one user.
  - **Optionality**: Mandatory for VoiceCommand, optional for User (a user may not have issued commands).
- **User to Device**:
  - **Relationship**: One-to-Many
  - **Description**: A user can own multiple devices, with each device linked to one user.
  - **Optionality**: Mandatory for Device, optional for User.
- **VoiceCommand to Interaction**:
  - **Relationship**: One-to-Many
  - **Description**: A voice command can be used in multiple interactions, with each interaction tied to one command.
  - **Optionality**: Mandatory for Interaction, optional for VoiceCommand.
- **Skill to Interaction**:
  - **Relationship**: One-to-Many
  - **Description**: A skill can be involved in multiple interactions, with each interaction involving one skill.
  - **Optionality**: Optional for Skill, mandatory for Interaction.
- **User to Skill (via UserSkill)**:
  - **Relationship**: Many-to-Many
  - **Description**: Users can enable multiple skills, and each skill can be used by multiple users.
  - **Optionality**: Optional for both User and Skill.
- **Device to VoiceCommand (via DeviceCommand)**:
  - **Relationship**: Many-to-Many
  - **Description**: A device can execute multiple commands, and a command can be executed on multiple devices.
  - **Optionality**: Optional for both Device and VoiceCommand.
- **User to Device (One-to-One)**:
  - **Relationship**: One-to-One
  - **Description**: Each user has a primary device.
  - **Optionality**: Mandatory for both User and Device.
- **Skill to SkillUsage**:
  - **Relationship**: One-to-Many
  - **Description**: A skill can have multiple usage records, each tied to a skill.
  - **Optionality**: Mandatory for SkillUsage, optional for Skill.

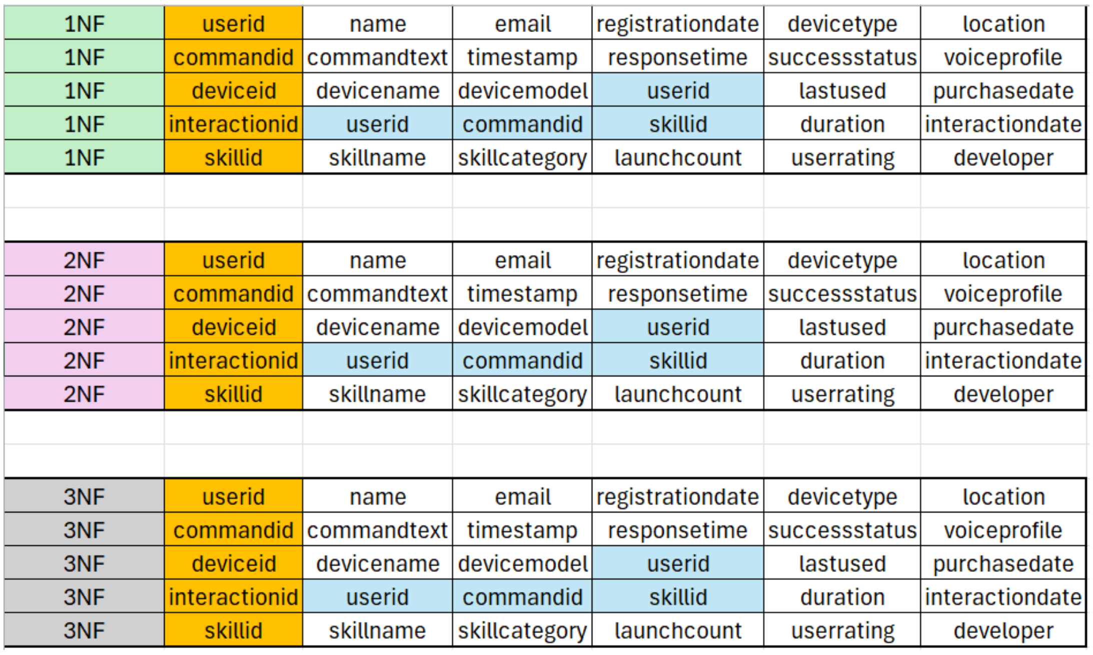
3. Assumptions and Special Considerations (5 Points)

- **Assumptions**:
    - All users must have at least one registered device.
    - Every voice command must be linked to a specific user, ensuring each interaction is user-specific.
    - Skills can be enabled or disabled based on user preference, and interactions are only recorded when skills are enabled.
- **Special Considerations**:
    - User data privacy is a priority, with secure storage and limited access to sensitive information.
    - The schema accommodates scenarios where users may not have linked skills or devices, or may not have issued commands, ensuring flexible usage.

4. Normalization Steps (20 Points)

The database design follows normalization principles to reduce redundancy and improve efficiency:

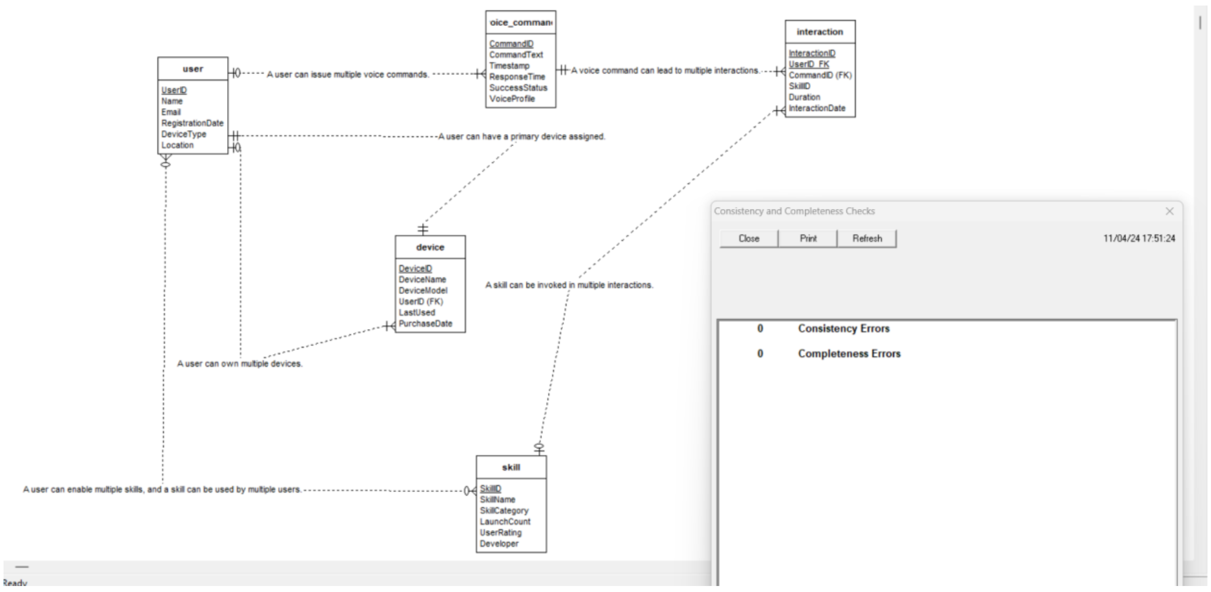
- **1NF (First Normal Form)**: All entities use atomic attributes, ensuring no repeating groups. For example, User has individual fields for Name and Email rather than concatenated attributes.
- **2NF (Second Normal Form)**: Each non-key attribute fully depends on the primary key. For instance, DeviceType in User relies solely on UserID, with no partial dependencies.
- **3NF (Third Normal Form)**: Transitive dependencies are removed by creating junction tables for many-to-many relationships, such as UserSkill and DeviceCommand, ensuring that no non-key attribute indirectly depends on another.



1NF	userid	name	email	registrationdate	devicetype	location
1NF	commandid	commandtext	timestamp	responsetime	successstatus	voiceprofile
1NF	deviceid	devicename	devicemodel	userid	lastused	purchasedate
1NF	interactionid	userid	commandid	skillid	duration	interactiondate
1NF	skillid	skillname	skillcategory	launchcount	userrating	developer
2NF	userid	name	email	registrationdate	devicetype	location
2NF	commandid	commandtext	timestamp	responsetime	successstatus	voiceprofile
2NF	deviceid	devicename	devicemodel	userid	lastused	purchasedate
2NF	interactionid	userid	commandid	skillid	duration	interactiondate
2NF	skillid	skillname	skillcategory	launchcount	userrating	developer
3NF	userid	name	email	registrationdate	devicetype	location
3NF	commandid	commandtext	timestamp	responsetime	successstatus	voiceprofile
3NF	deviceid	devicename	devicemodel	userid	lastused	purchasedate
3NF	interactionid	userid	commandid	skillid	duration	interactiondate
3NF	skillid	skillname	skillcategory	launchcount	userrating	developer

This normalization process optimizes data integrity, eliminates redundancy, and enhances query performance in the Alexa interaction database.

5. ER Diagram:



11/04/24 17:51:24

```
0    Consistency Errors
0    Completeness Errors
```